WEEK 5

Model Evaluation

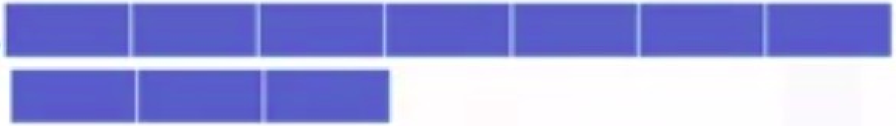
Model Evaluation

- In-sample evaluation tells us how well our model will fit the data used to train it
- Problem?
  - It does not tell us how well the trained model can be used to predict new data
- Solution?
  - In- sample data or training data
  - Out-of-sample evaluation or test set

Training/Testing Sets

Data:

- Split dataset into:
  - Training set (70%),
  - Testing set (30%)



- Build and train the model with a training set
- Use testing set to assess the performance of a predictive model
- When we have completed testing our model we should use all the data to train the model to get the best performance

Question

consider the following lines of code:

```
from sklearn.model_selection import train_test_split


x_train, x_test, y_train, y_test = train_test_split(x_data, y_data, test_size=0.3, random_state=0)
```

what does the variable **y_data** contain

- ○ features or independent variables
- ○ percentage of the data for testing
- ◉ dataset target

Correct
correct

Skip Continue

Function train_test_split()

- Split data into random train and test subsets

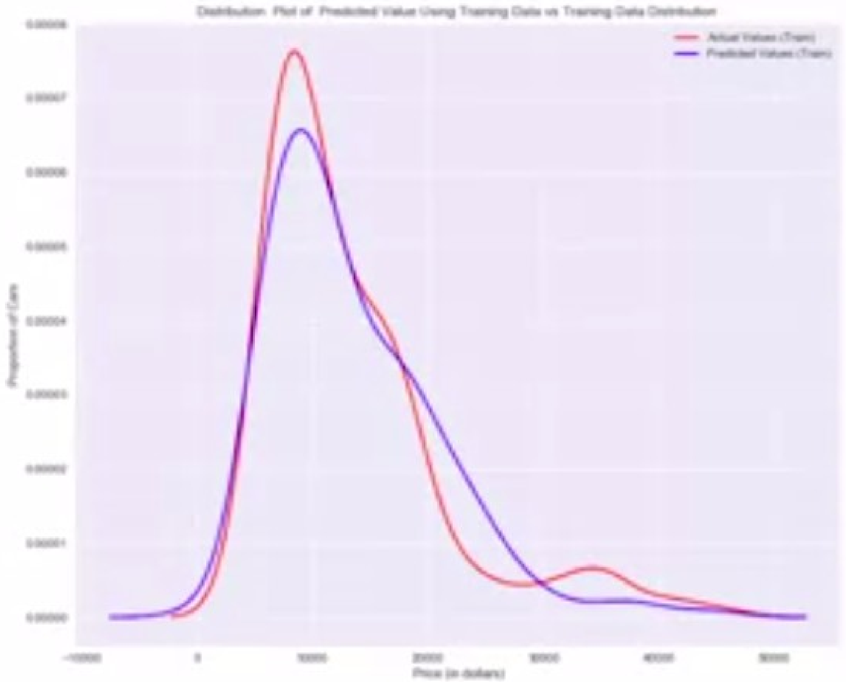
```
from sklearn.model_selection import train_test_split

x_train, x_test, y_train, y_test = train_test_split(x_data, y_data, test_size=0.3, random_state=0)
```

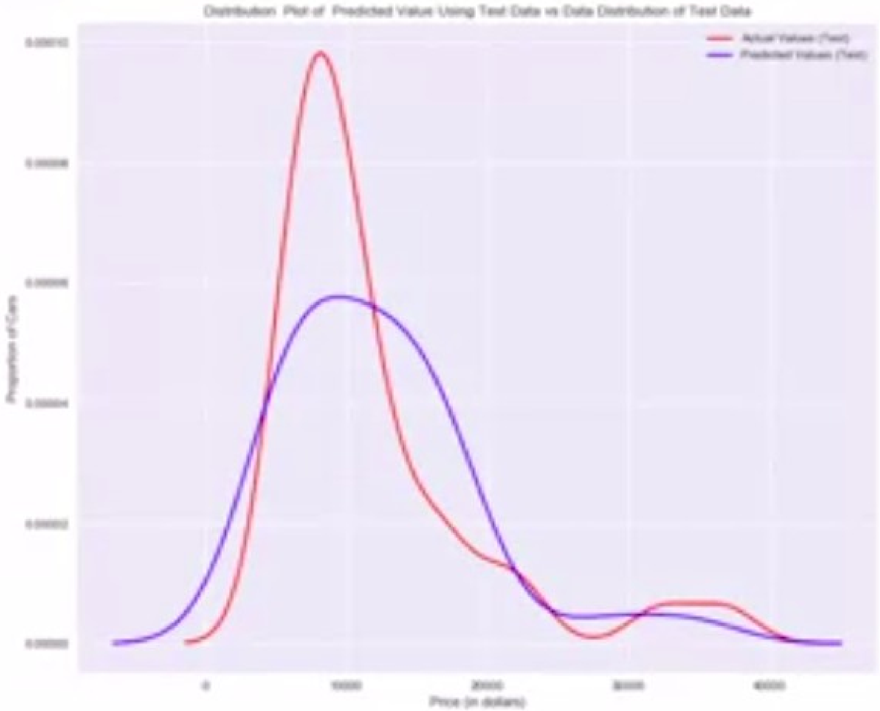
- **x_data**: features or independent variables
- **y_data**: dataset target: df['price']
- **x_train, y_train**: parts of available data as training set
- **x_test, y_test**: parts of available data as testing set
- **test_size**: percentage of the data for testing (here 30%)
- **random_state**: number generator used for random sampling

Generalization Performance

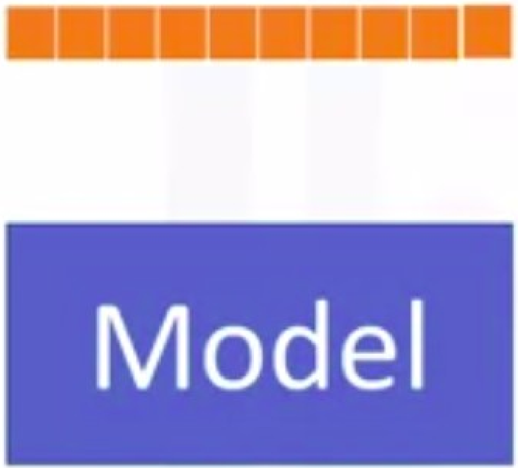
- Generalization error is measure of how well our data does at predicting previously unseen data
- The error we obtain using our testing data is an approximation of this error

Generalization Error

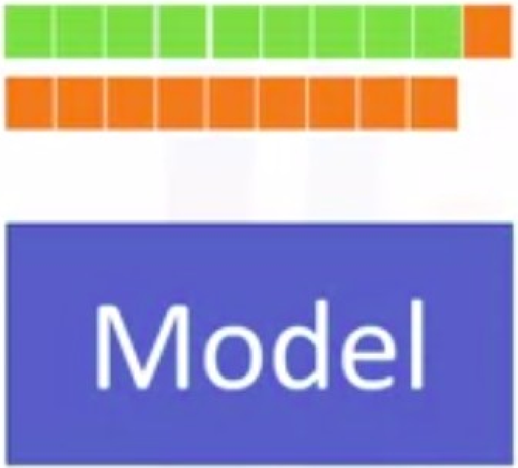


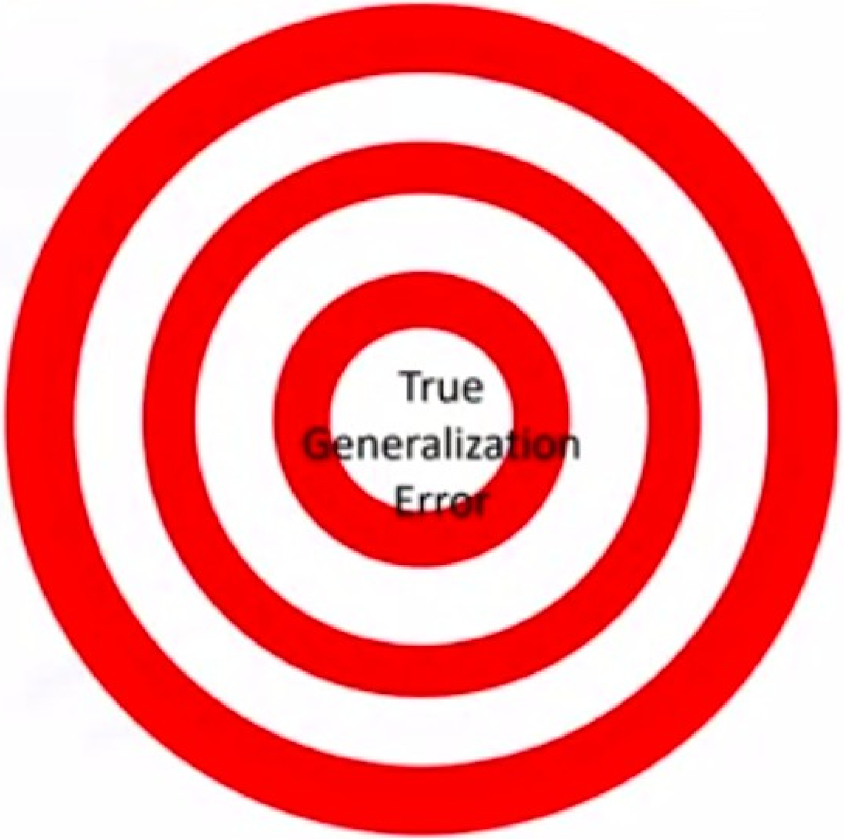


Lots of Training Data



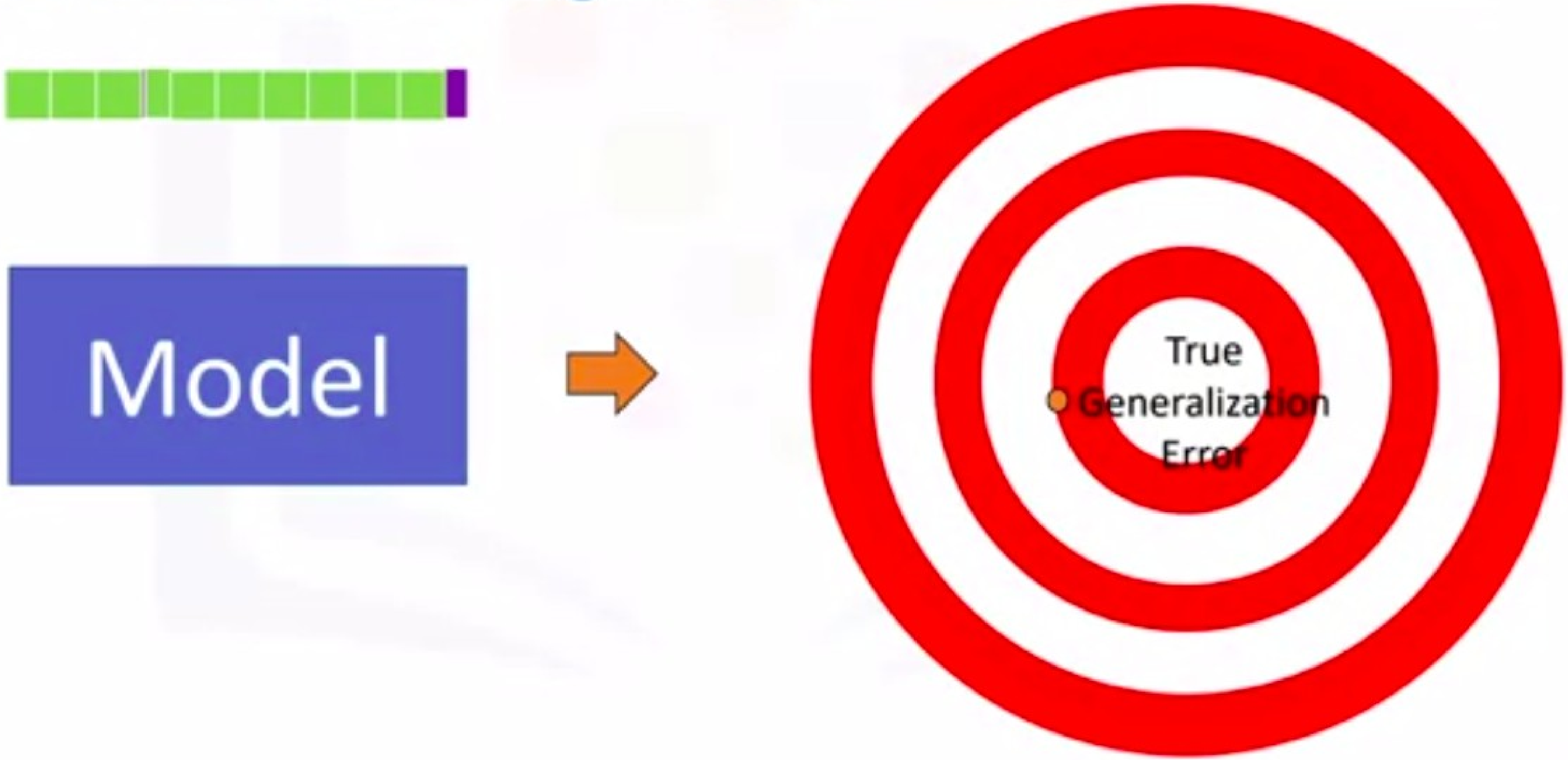
Lots of Training Data



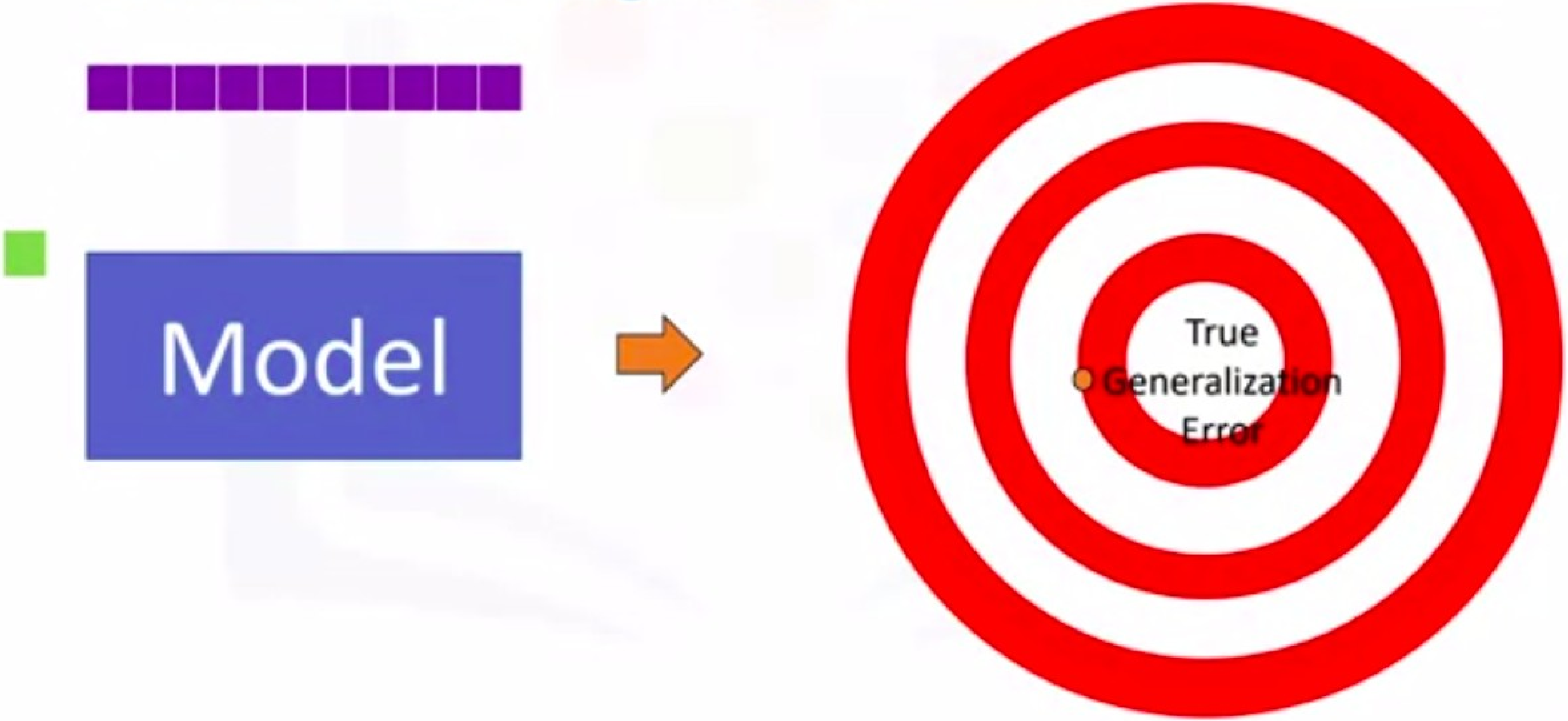


Lots of Training Data

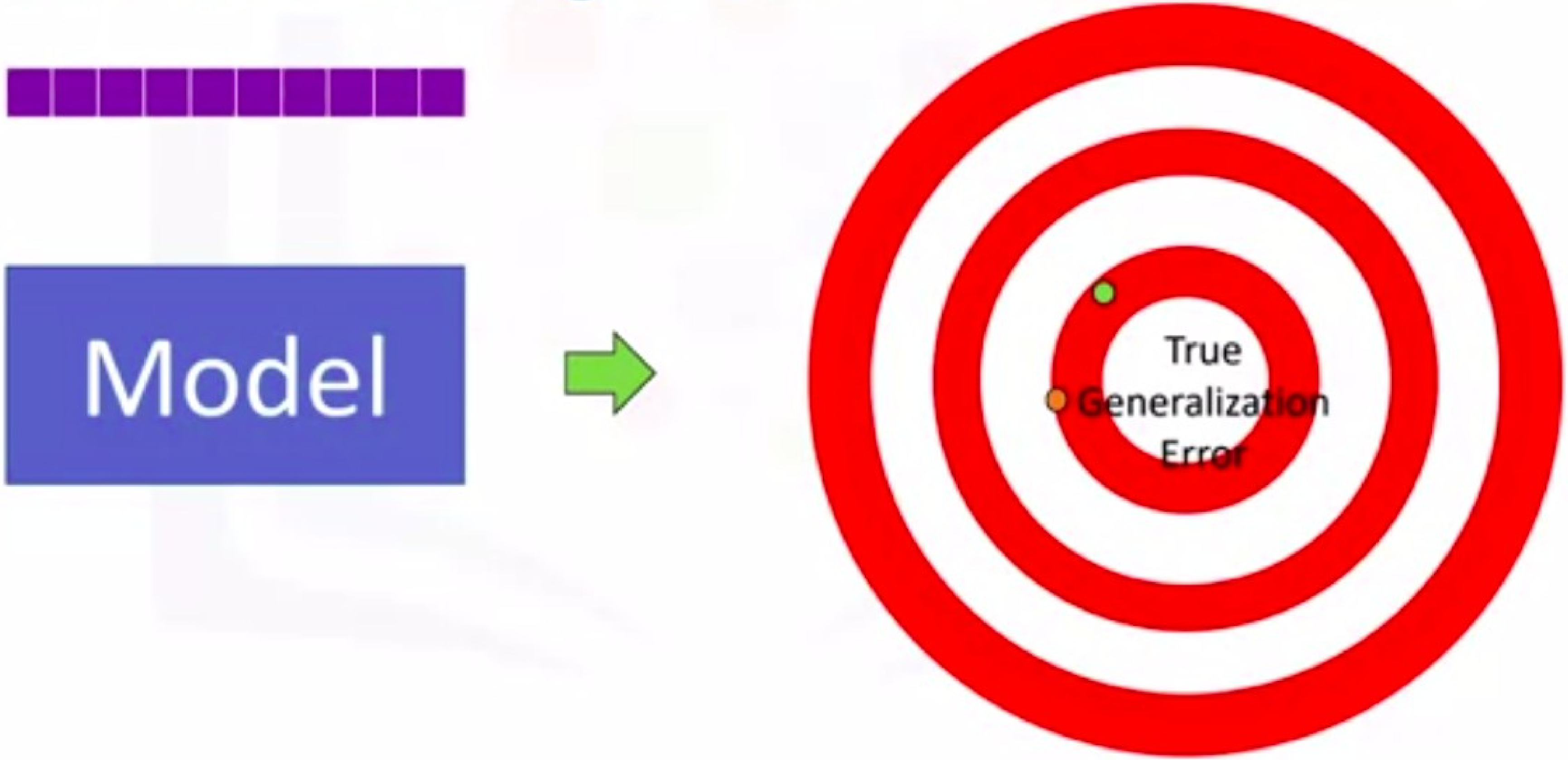
Lots of Training Data



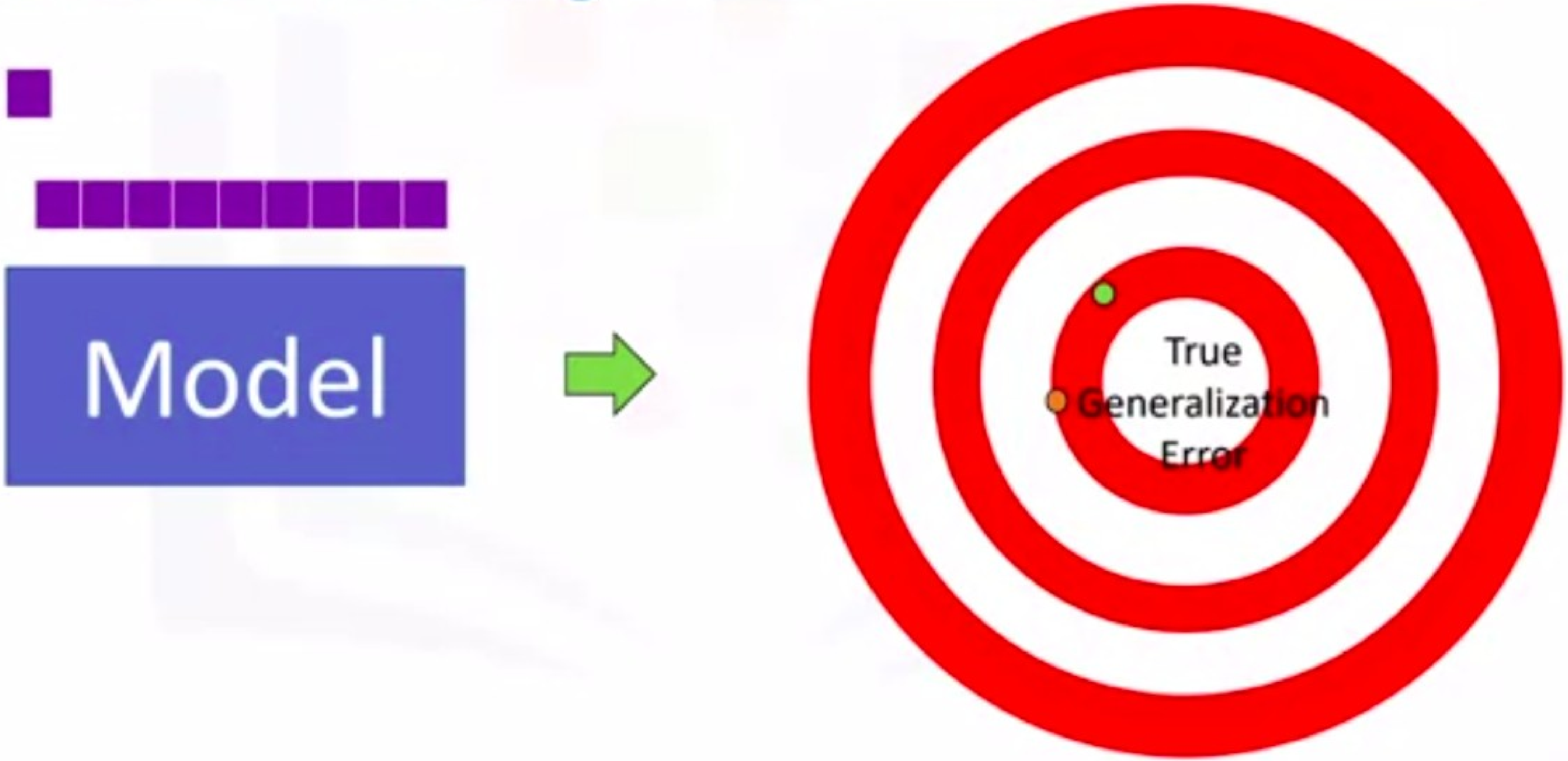
Lots of Training Data



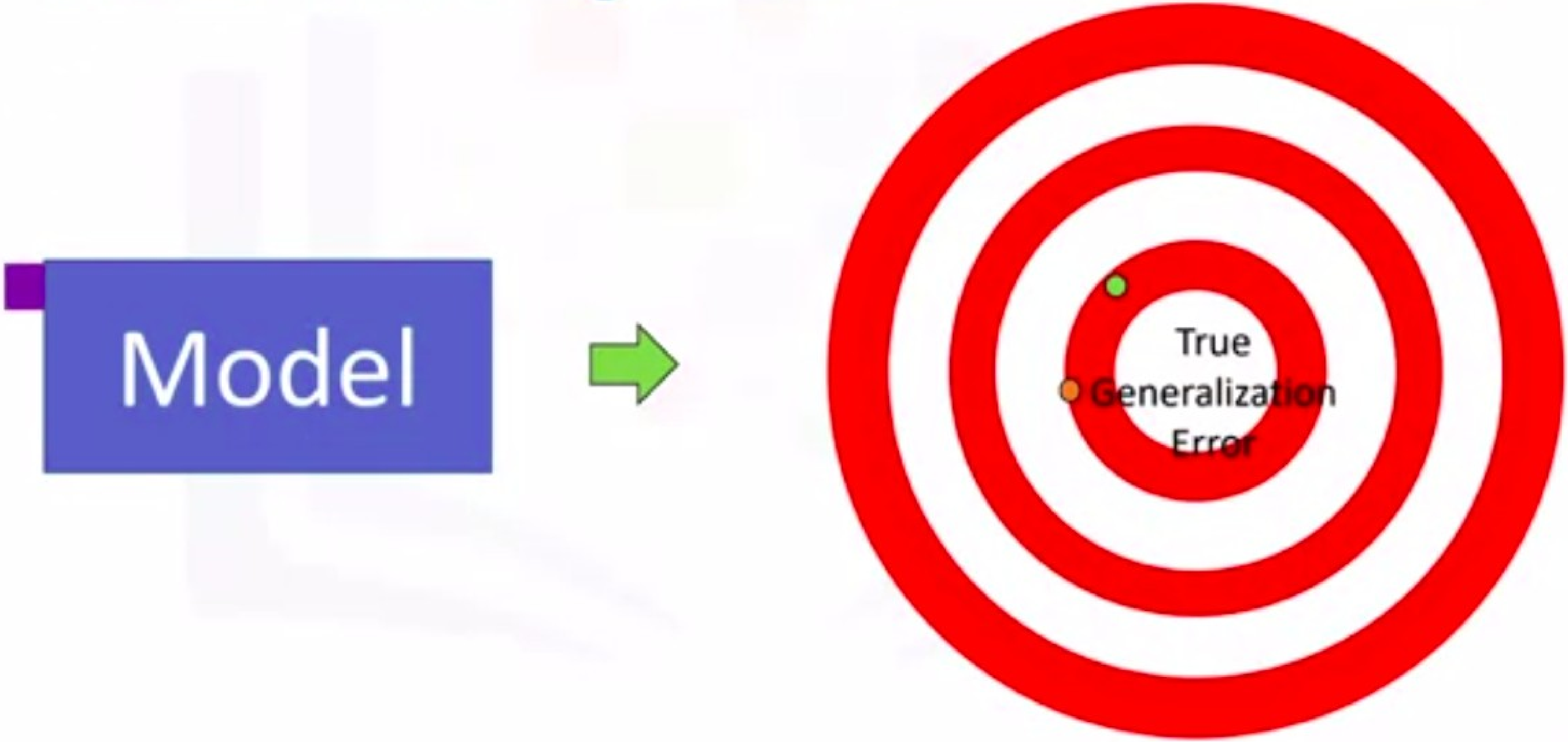
Lots of Training Data



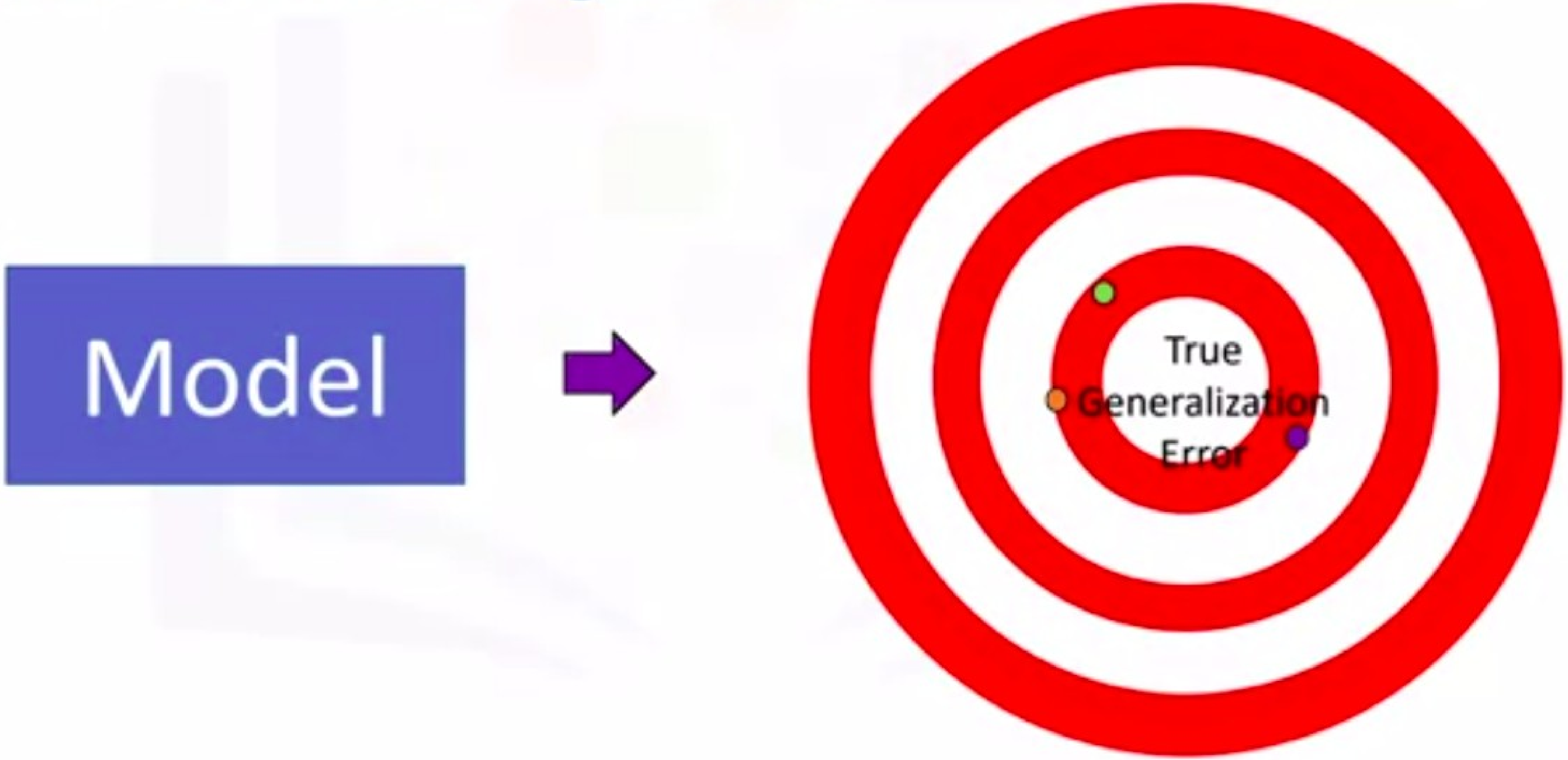
Lots of Training Data



Lots of Training Data

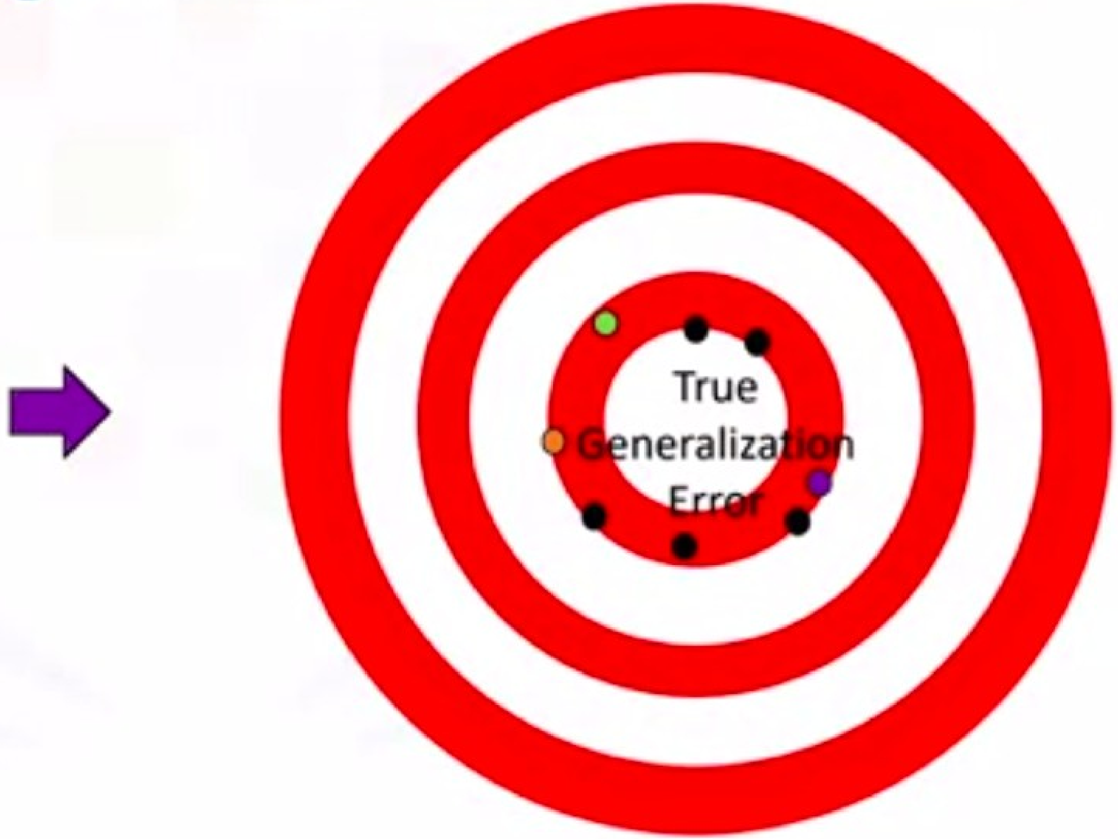


Lots of Training Data



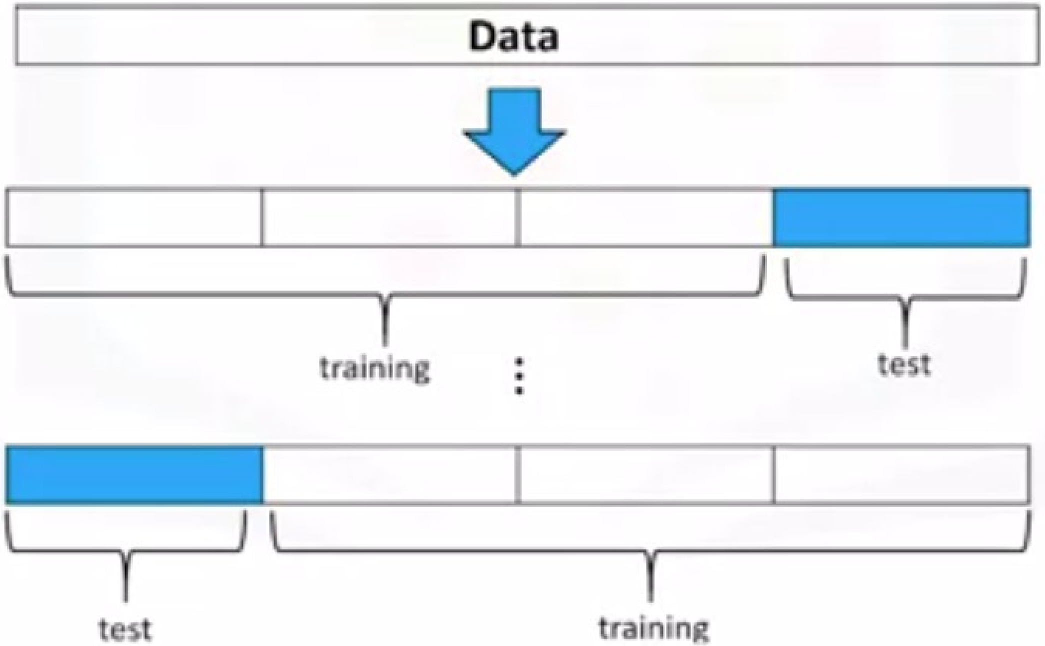
Lots of Training Data

Model



Cross Validation

- Most common out-of-sample evaluation metrics
- More effective use of data (each observation is used for both training and testing)



Question

consider the following lines of code, how may partitions or folds are used in the function **cross_val_score**:

```
from sklearn.model_selection import cross_val_sc
scores= cross_val_score(lr, x_data, y_data, cv=10)

```

- 4
- 10
- 5

Correct
correct

Skip

Continue

Function cross_val_score()

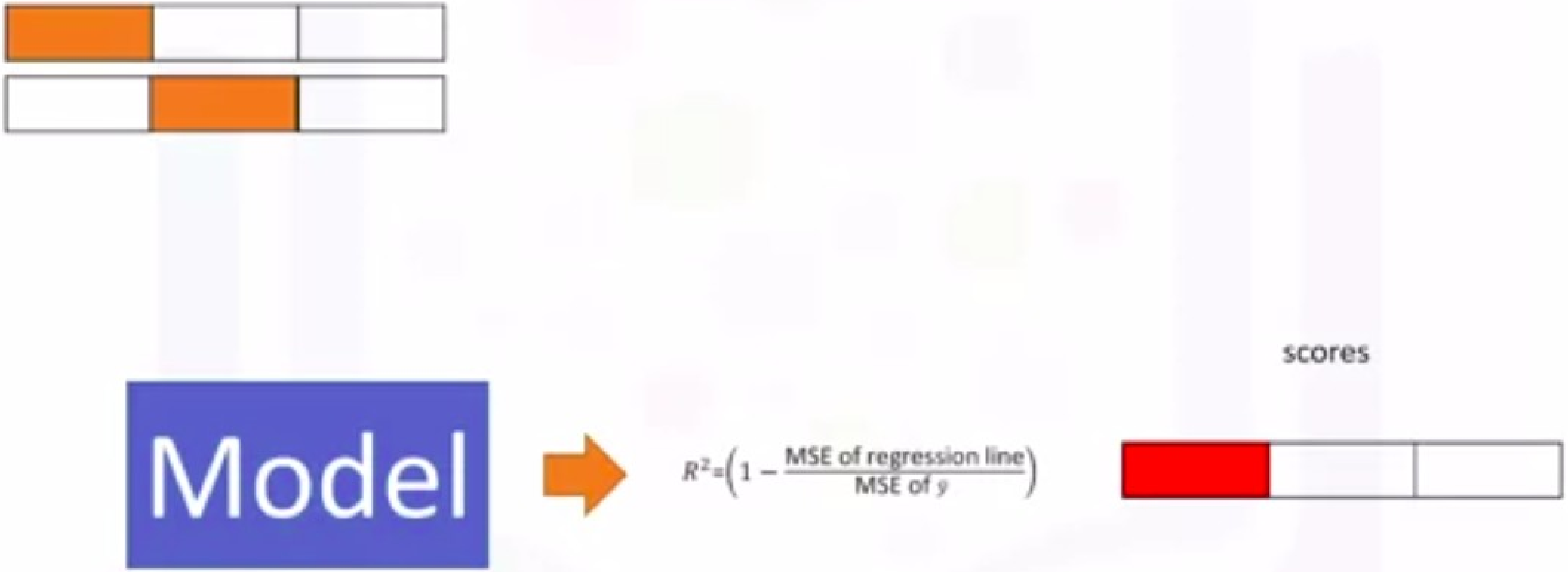
```
from sklearn.model_selection import cross_val_score

scores= cross_val_score(lr, x_data, y_data, cv=3)

np.mean(scores)
```

Function cross_val_score()

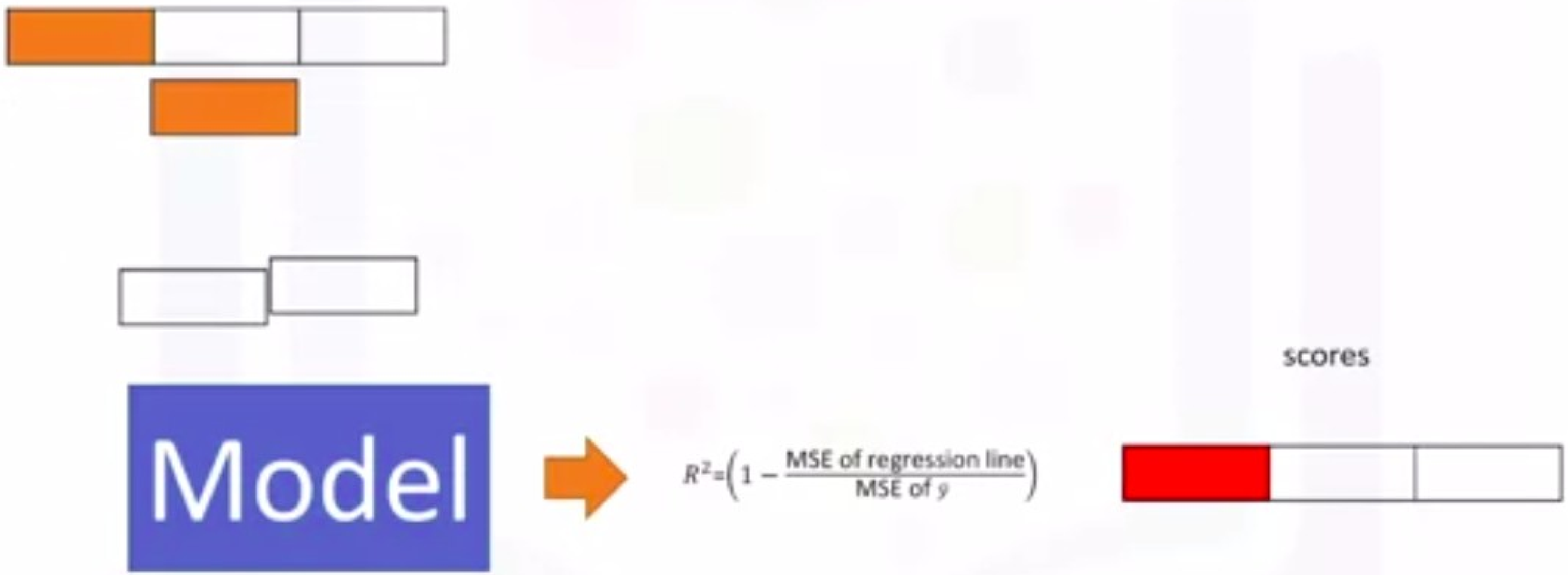
Function cross_val_score()

Function cross_val_score()

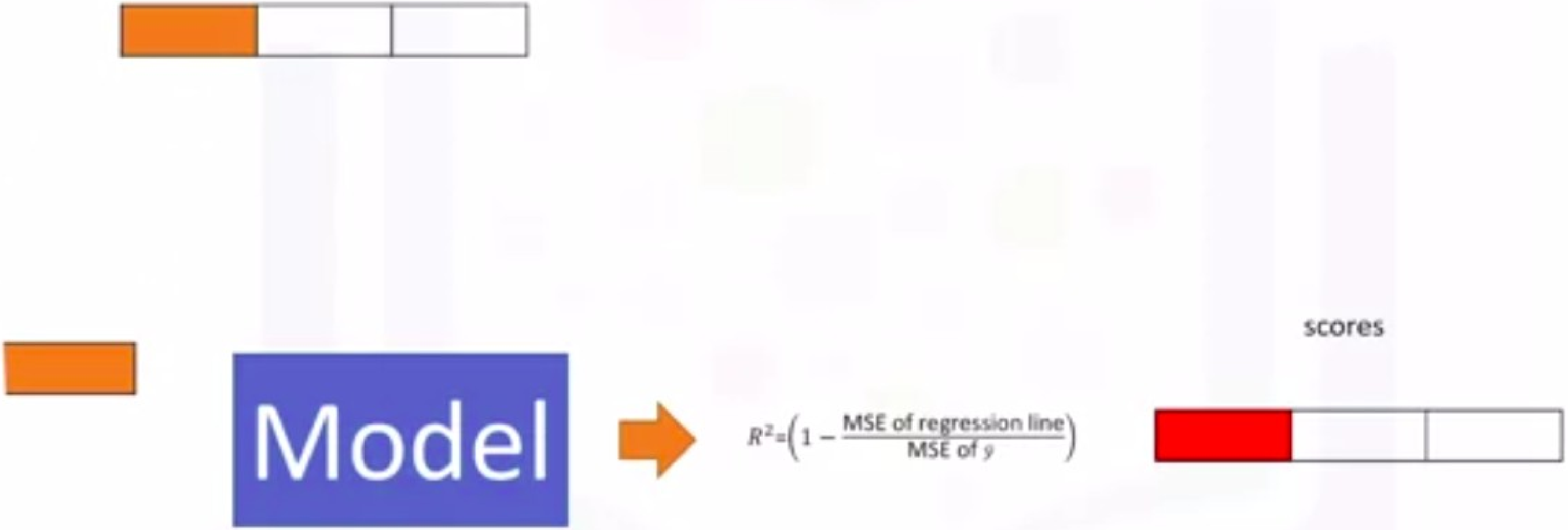
Function cross_val_score()



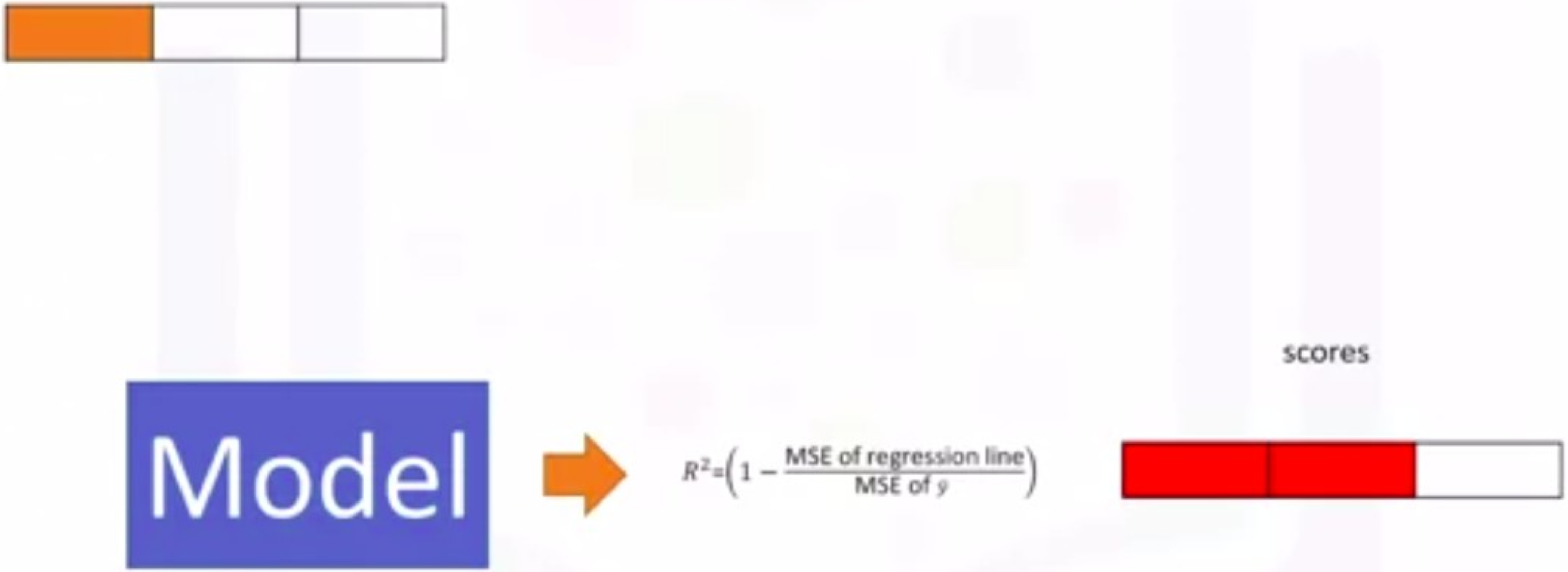
Function cross_val_score()



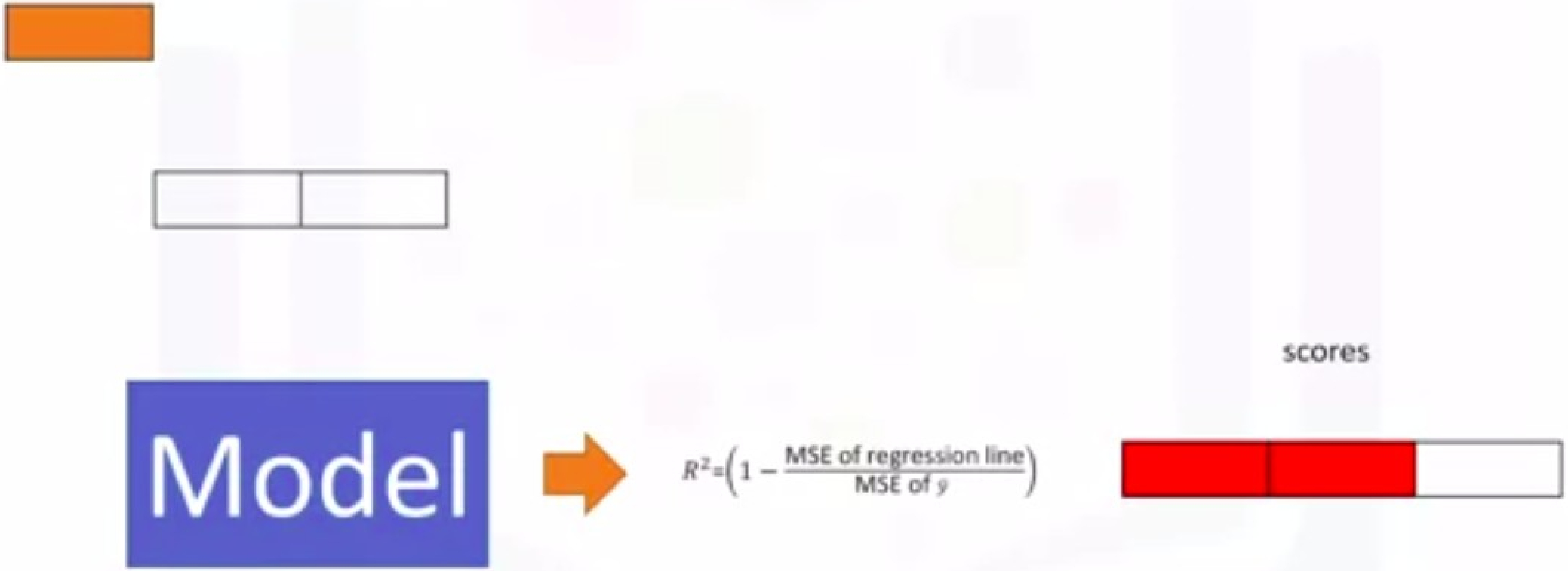
Function cross_val_score()



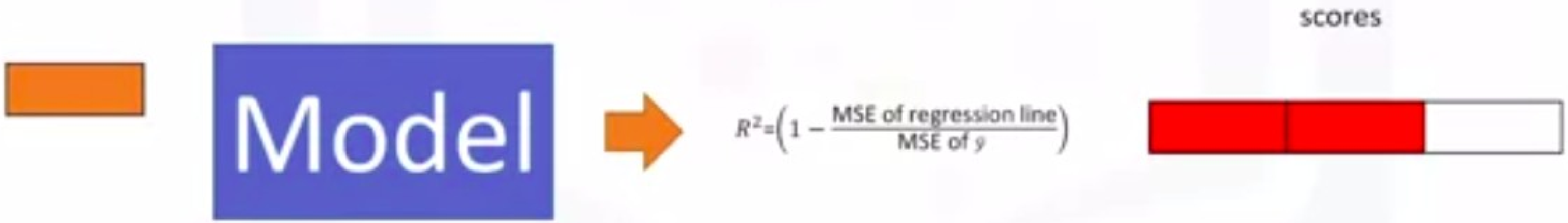
Function cross_val_score()



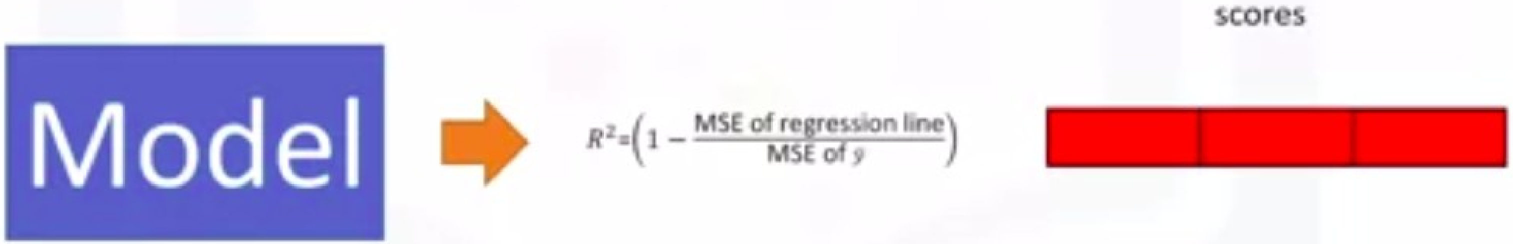
Function cross_val_score()



Function cross_val_score()



Function cross_val_score()

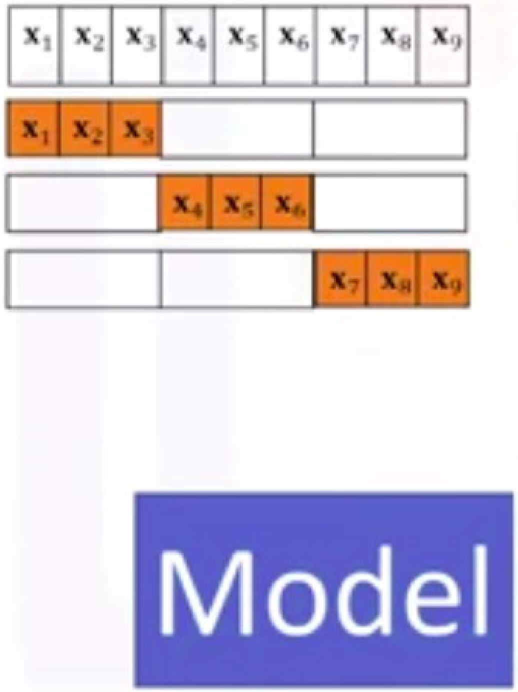


Function cross_val_predict()

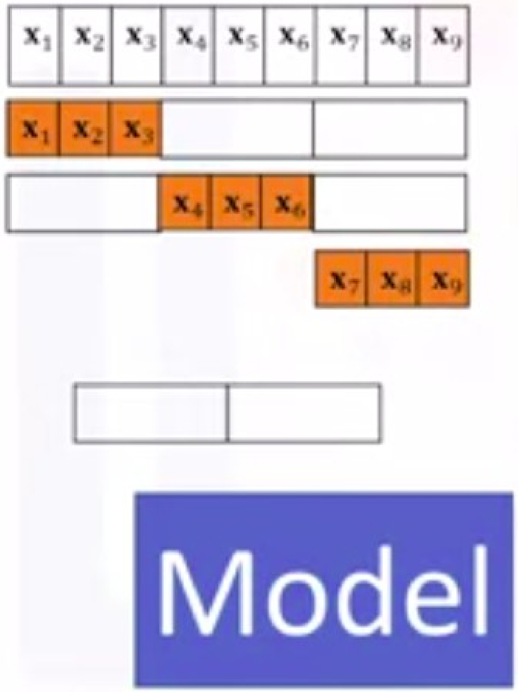
- It returns the prediction that was obtained for each element when it was in the test set
- Has a similar interface to cross_val_score()

```
from sklearn.model_selection import cross_val_predict

yhat= cross_val_predict (lr2e, x_data, y_data, cv=3)
```

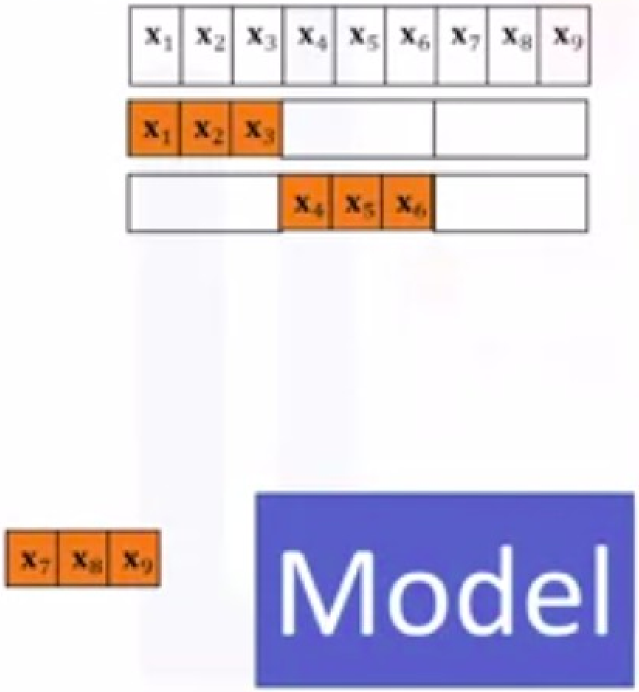
Function cross_val_predict()



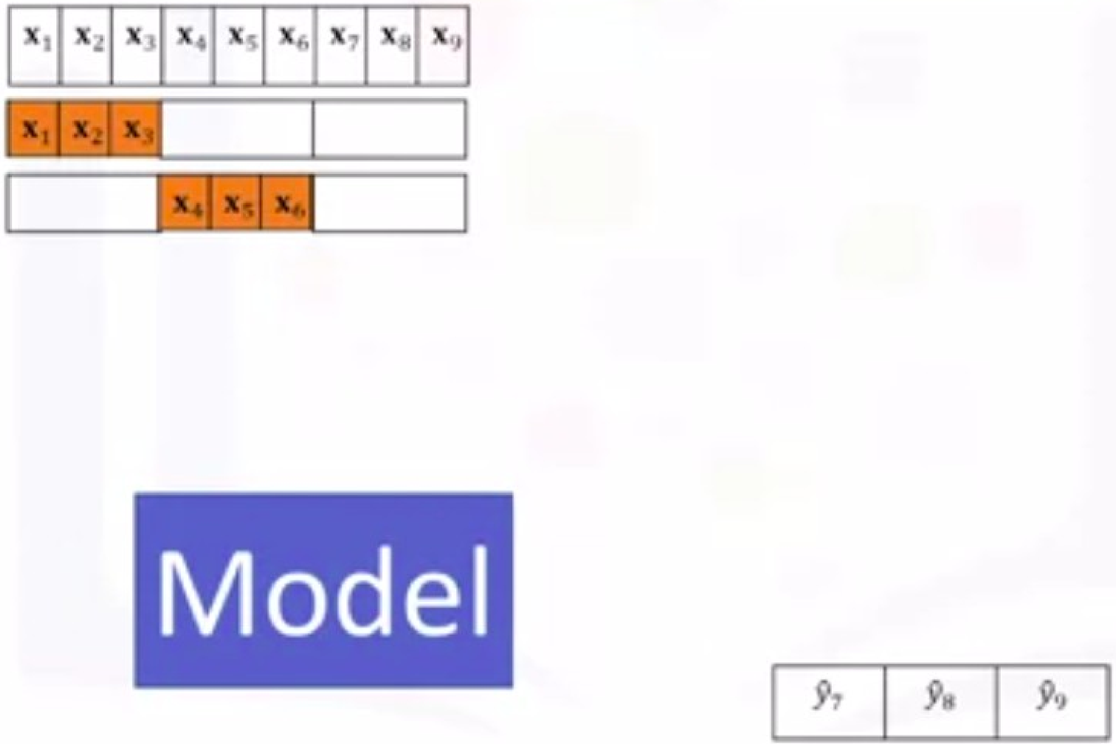
Function cross_val_predict()



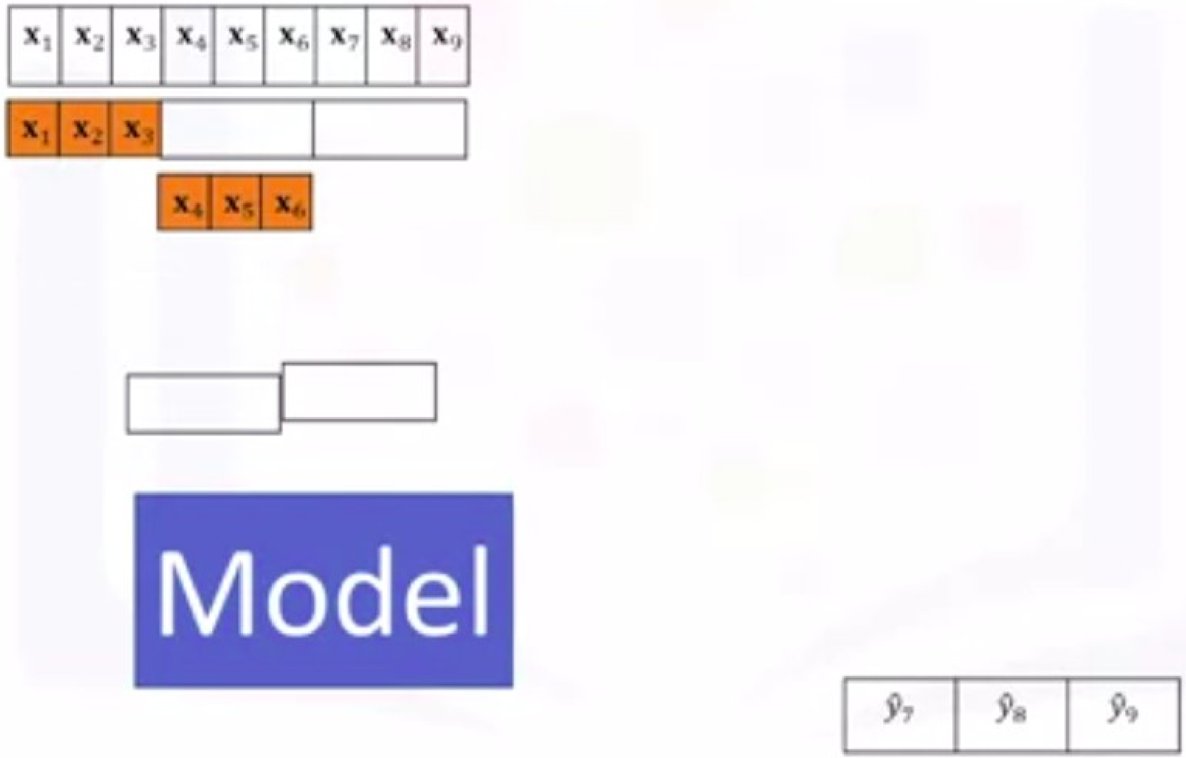
Function cross_val_predict()



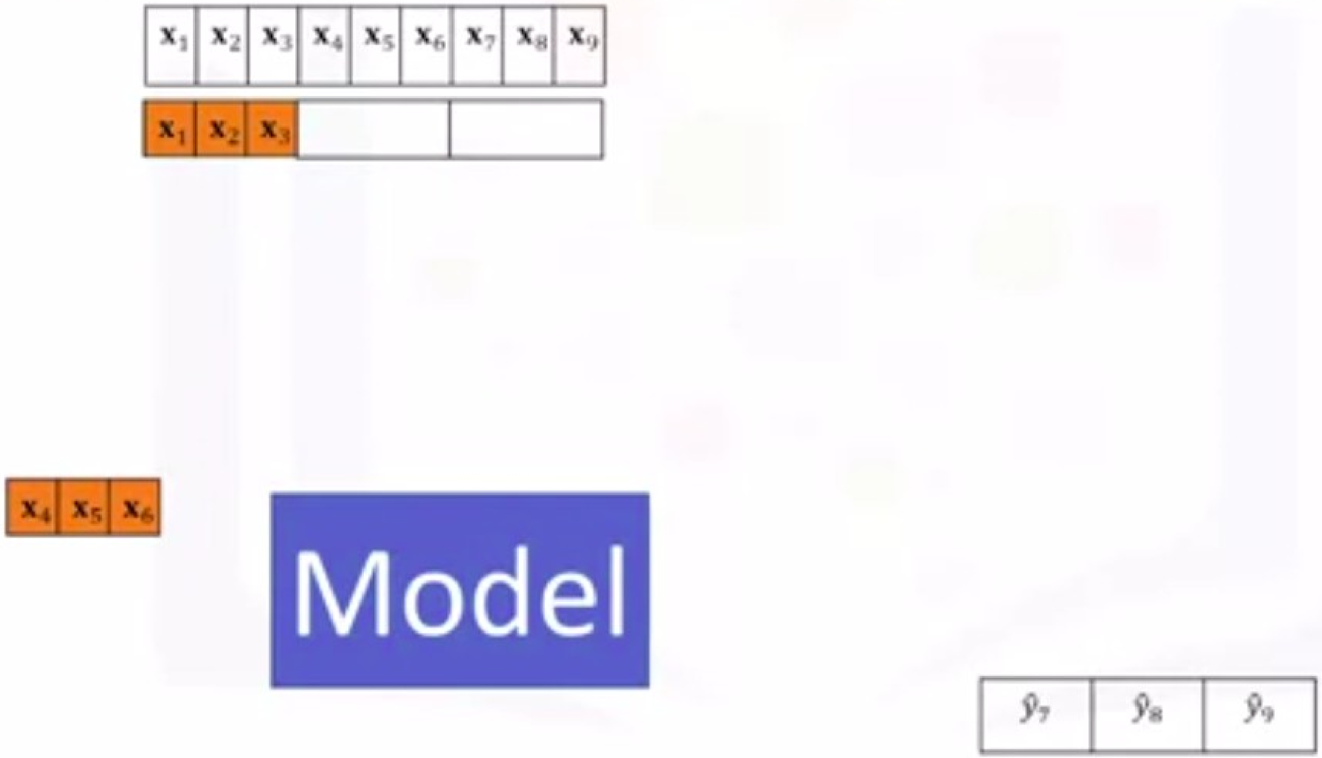
Function cross_val_predict()



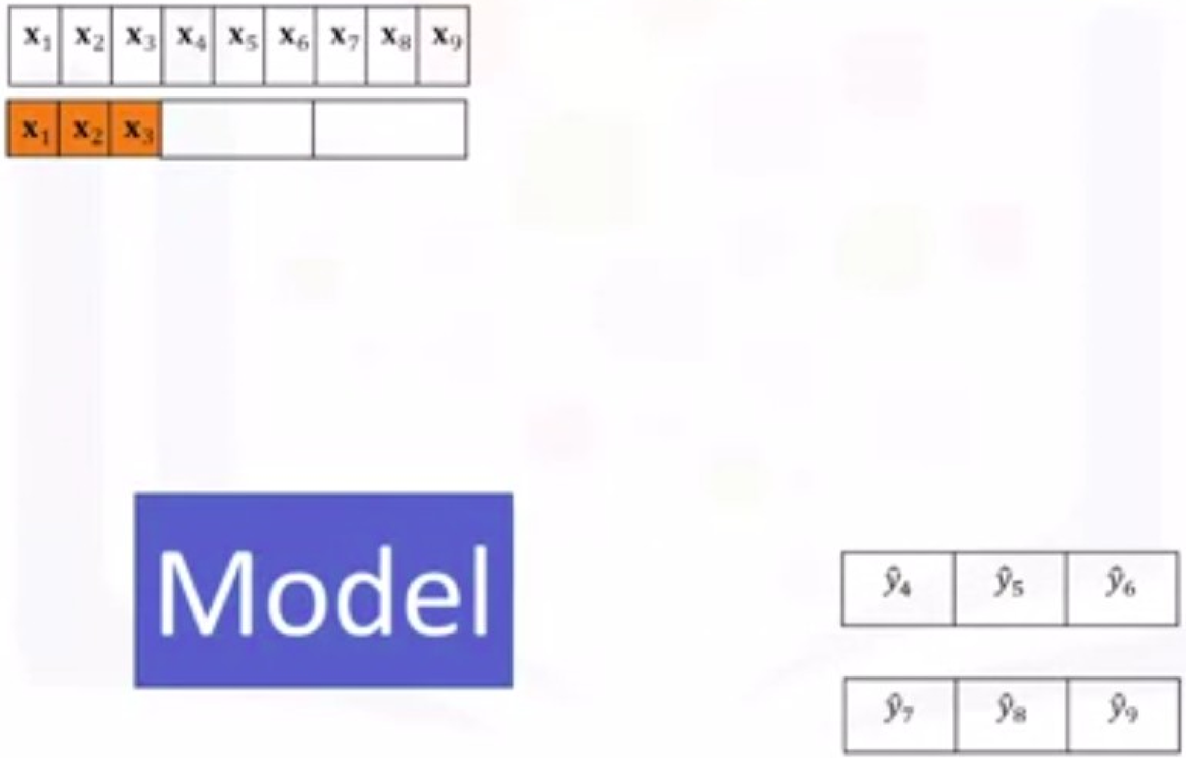
Function cross_val_predict()



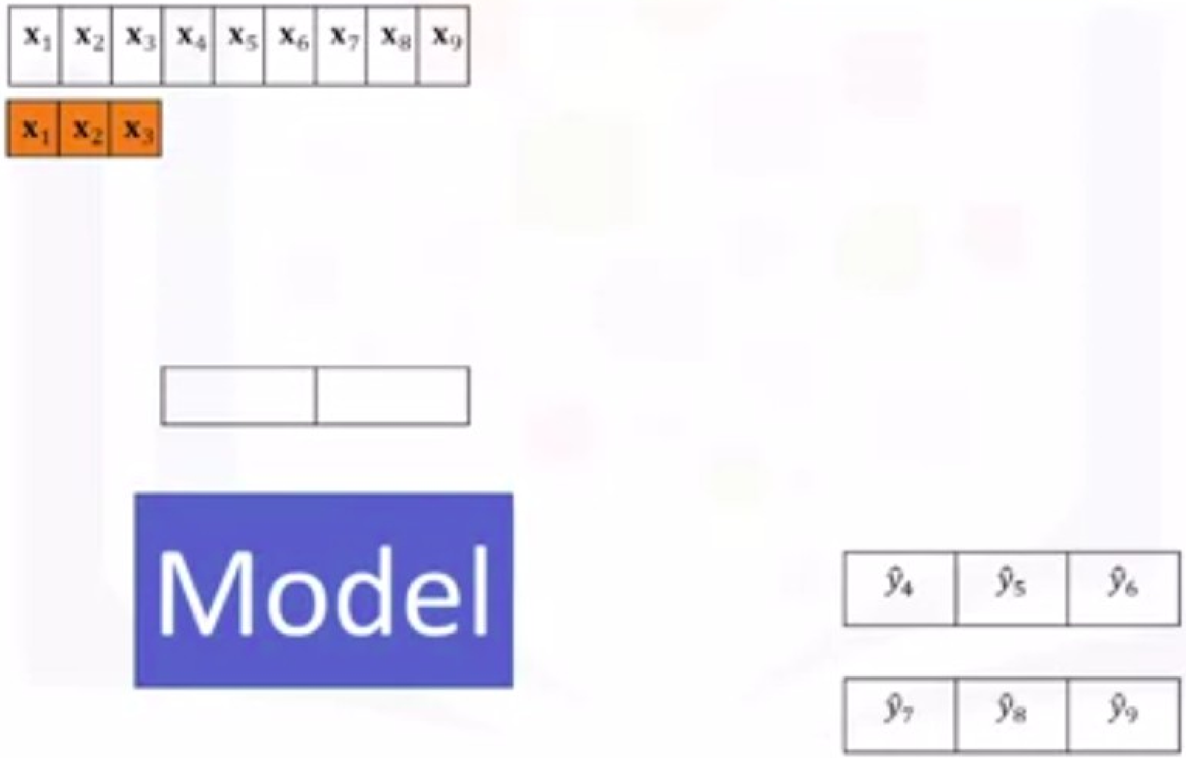
Function cross_val_predict()



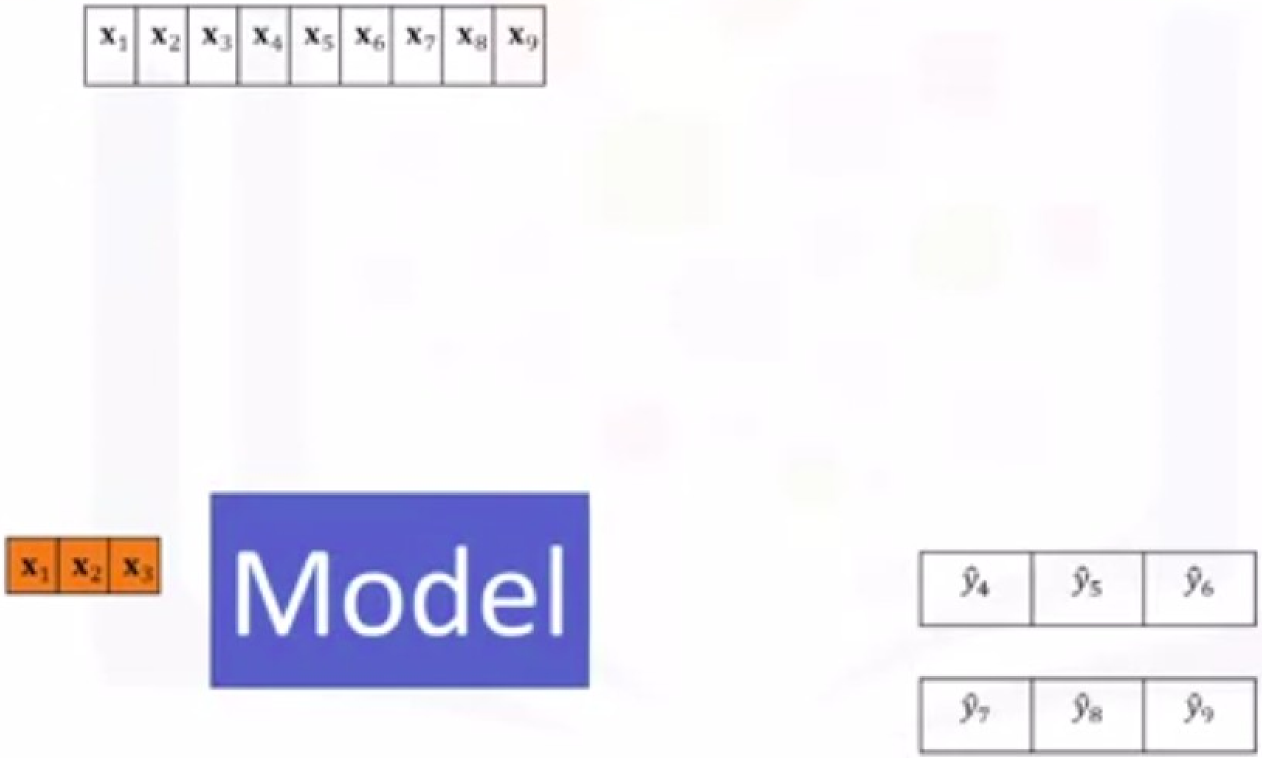
Function cross_val_predict()



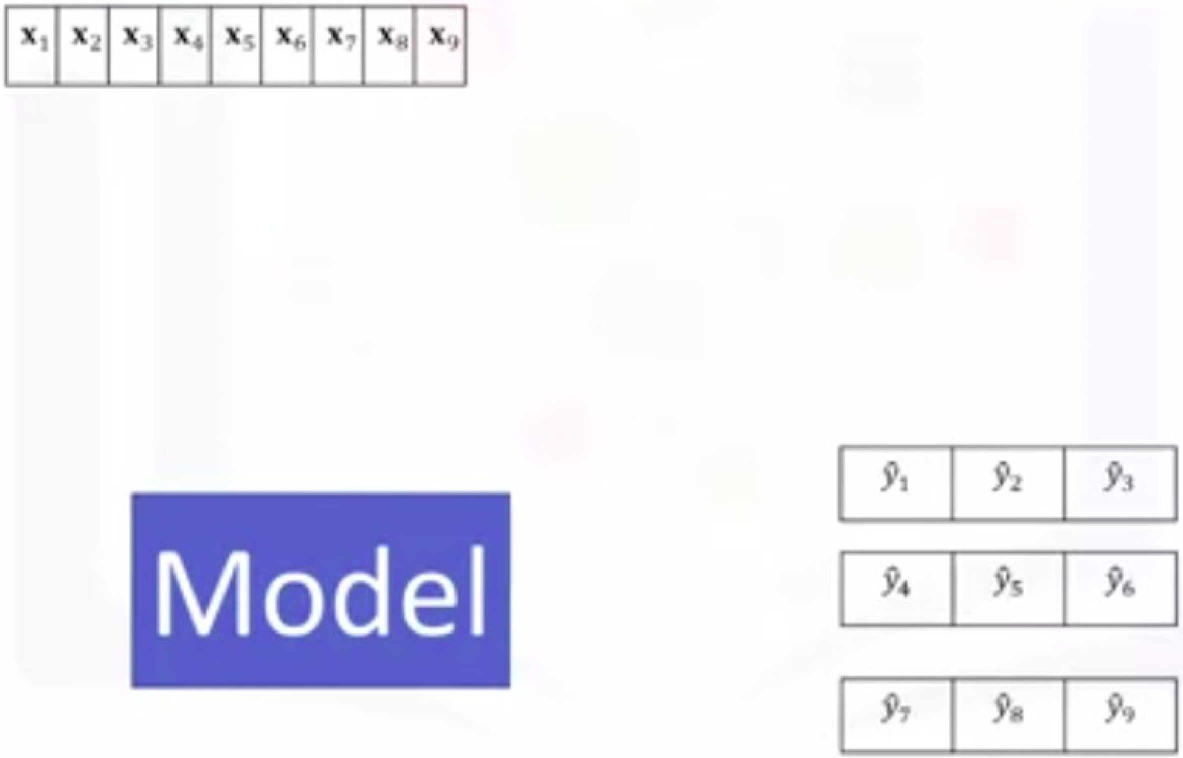
Function cross_val_predict()



Function cross_val_predict()



Function cross_val_predict()



Back

Practice Quiz: Model Evaluation

Practice Quiz • 3 min • 1 total point

Congratulations! You passed!

Grade received 100%   **To pass** 50% or higher

Go to next item

1. What is the correct use of the "train_test_split" function such that 90% of the data samples will be utilized for training, the parameter "random_state" is set to zero, and the input variables for the features and targets are x_data, y_data respectively.

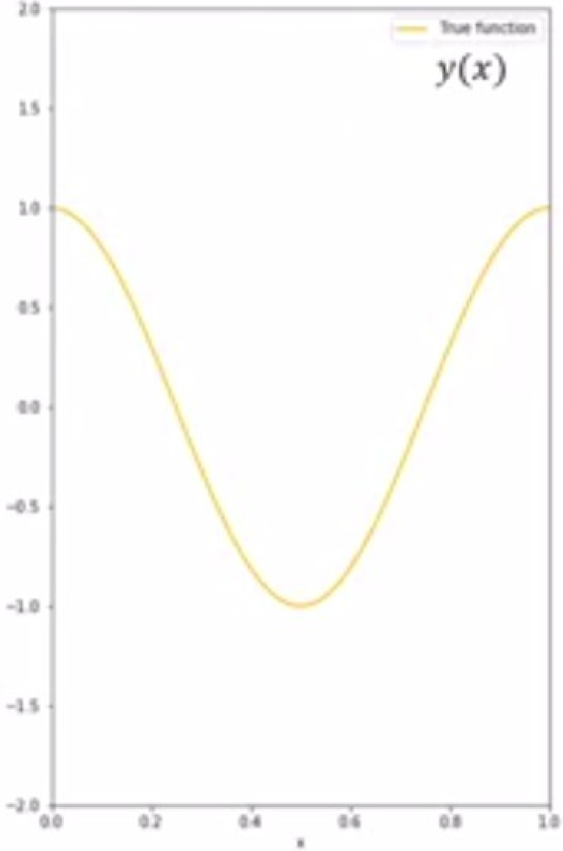
1 / 1 point

- [ ]
```
train_test_split(x_data, y_data, test_size=0.9, random_state=0)
```

- [x]
```
train_test_split(x_data, y_data, test_size=0.1, random_state=0)
```

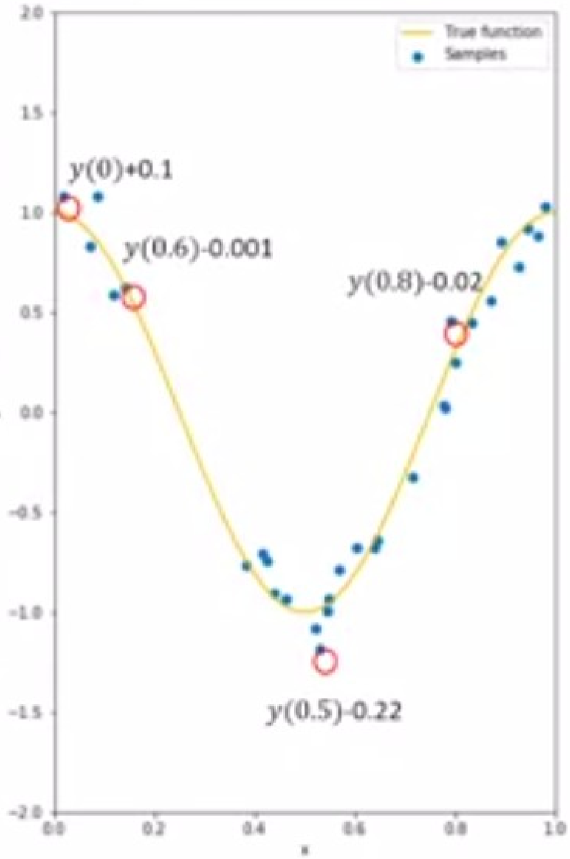
Overfitting, Underfitting and Model Selection

Model Selection

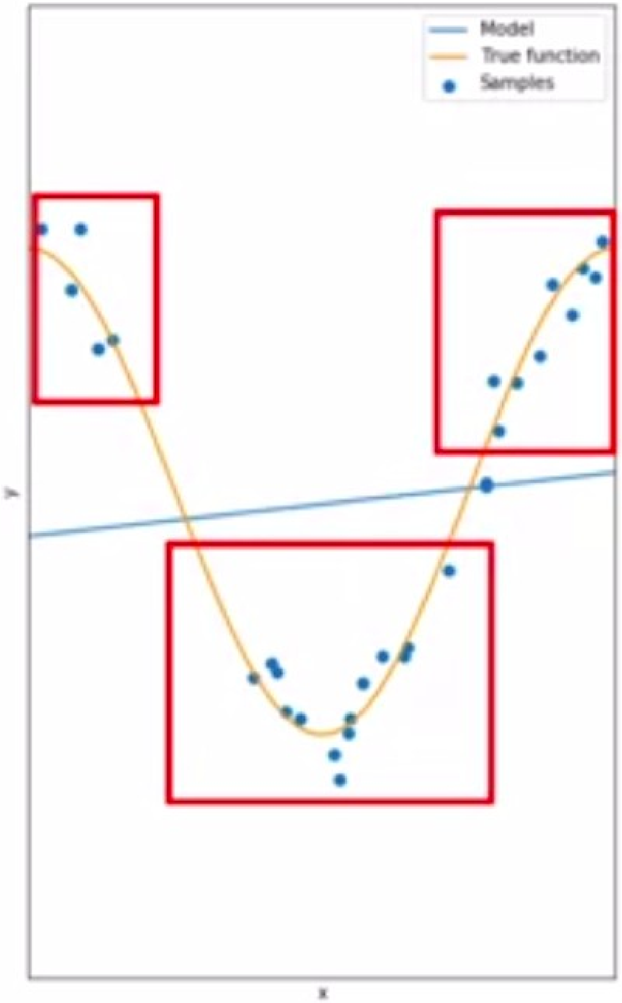


Model Selection

$y(x)$+noise



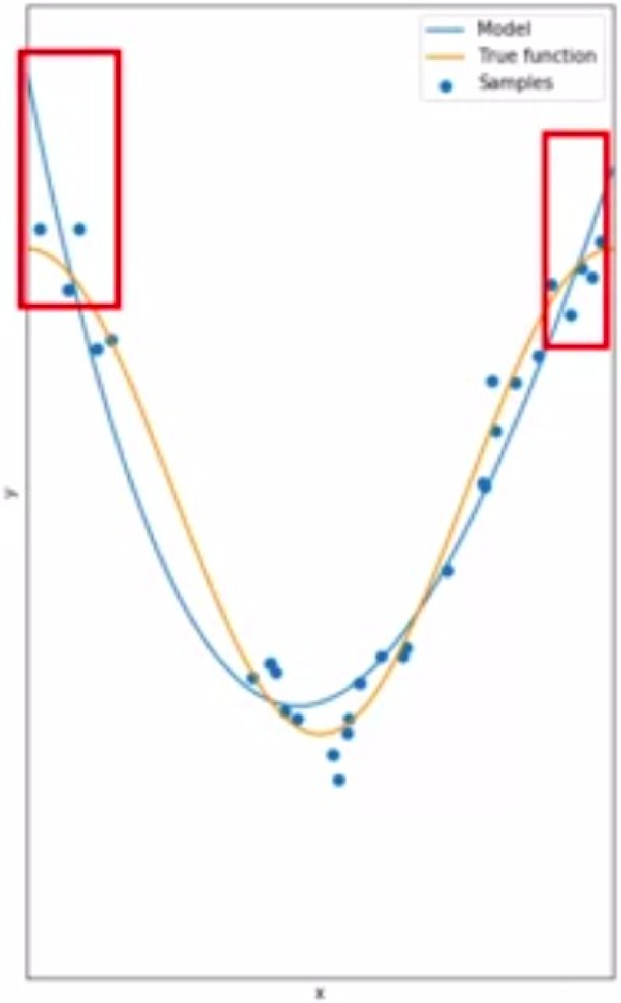
$$y = b_0 + b_1 \, x$$



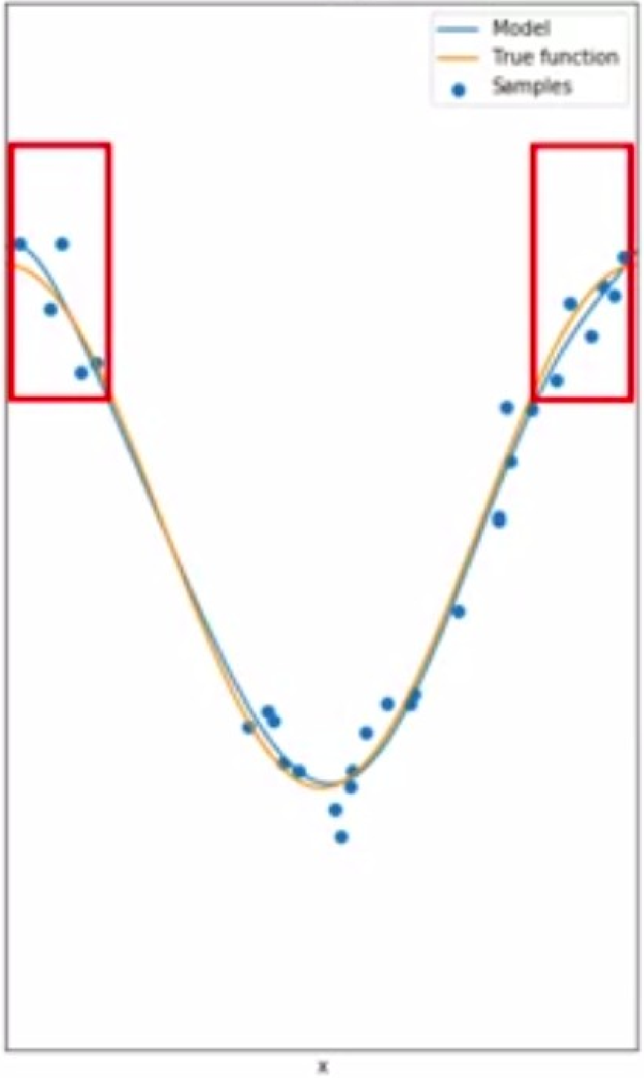
IBM Developer

SKILLS NETWORK

$$y = b_0 + b_1 x + b_2 x^2$$

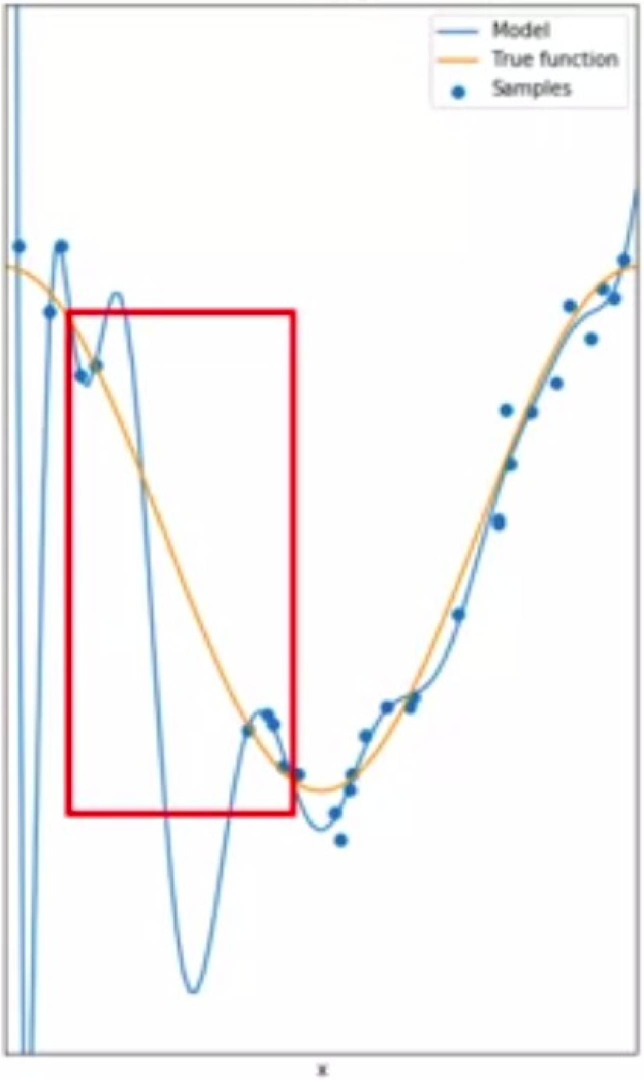


$$\hat{y} = b_0 + b_1 x + b_2 x^2 + b_3 x^3 + b_4 x^4 + b_5 x^5 + b_6 x^6 + b_7 x^7 + b_8 x^8$$



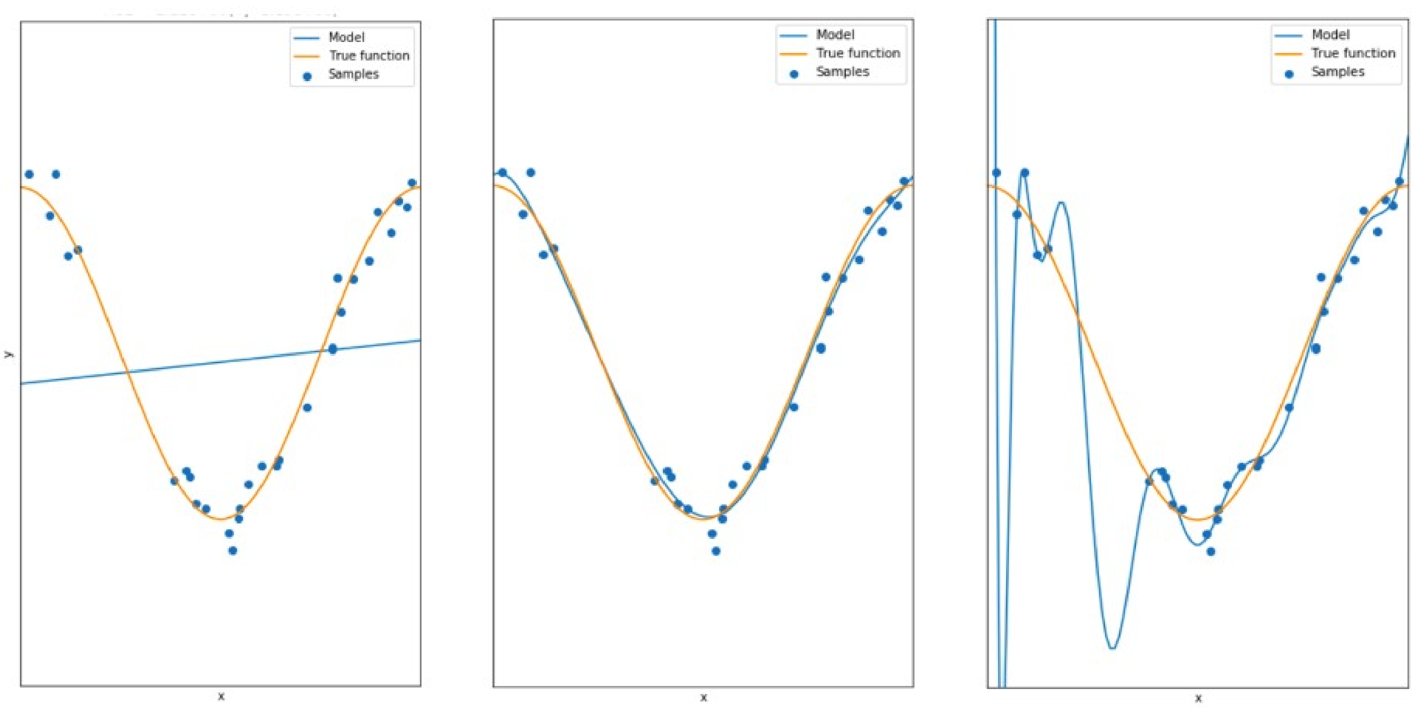
$$\hat{y} = b_0 + b_1 x + b_2 x^2 + b_3 x^3 + b_4 x^4 + b_5 x^5 + b_6 x^6 + b_7 x^7 + b_8 x^8 + ..$$

$$+ b_9 x^9 + b_{10} x^{10} + b_{11} x^{11} + b_{12} x^{12} + b_{13} x^{13} + b_{14} x^{14} + b_{15} x^{15} + b_{16} x^{16}$$



Question

select the plot a, b or c that best demonstrates overfitting:



Skip

Continue

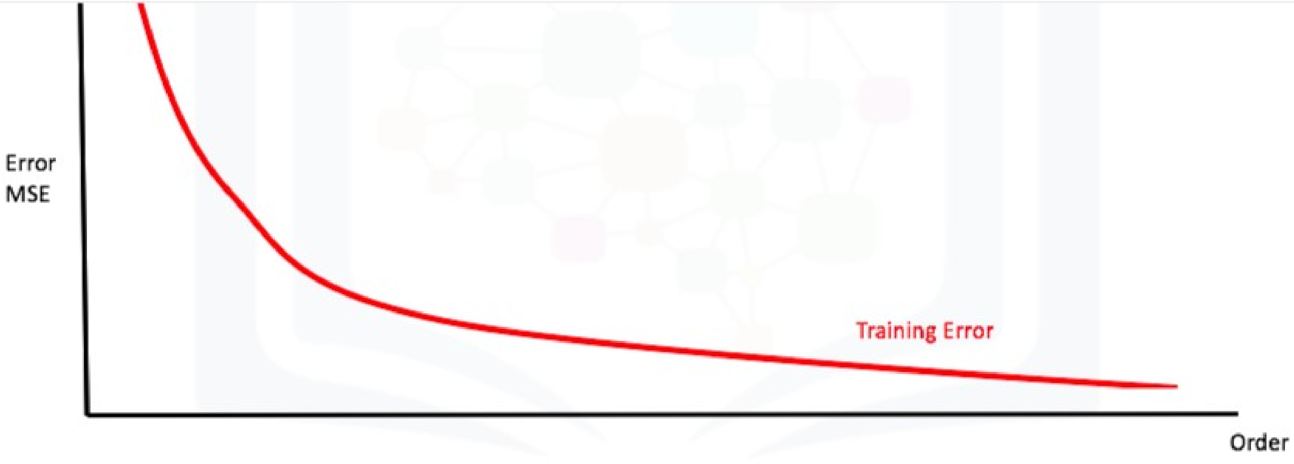
Question

True or False, the following plot shows that as the order of the polynomial increases the mean square error of our model decreases on the test data:

Skip

Continue

Question



- False
- True

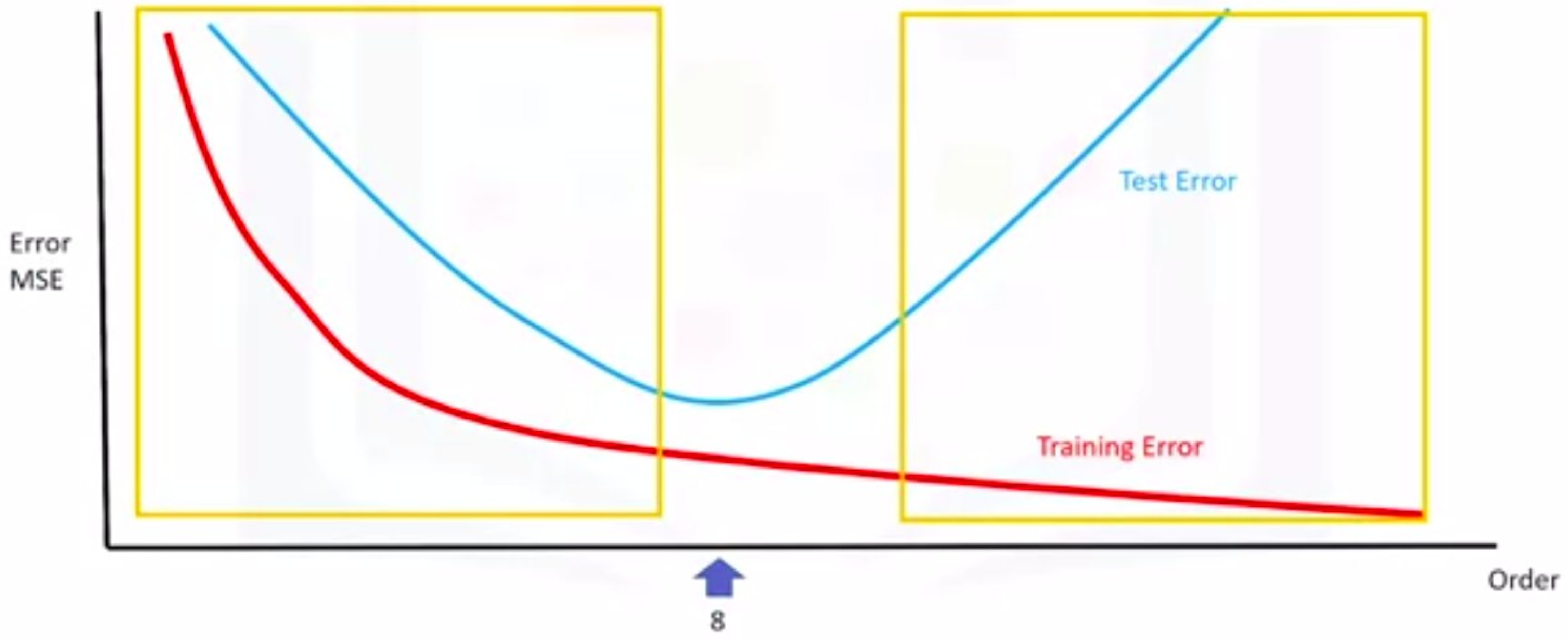
Correct

correct

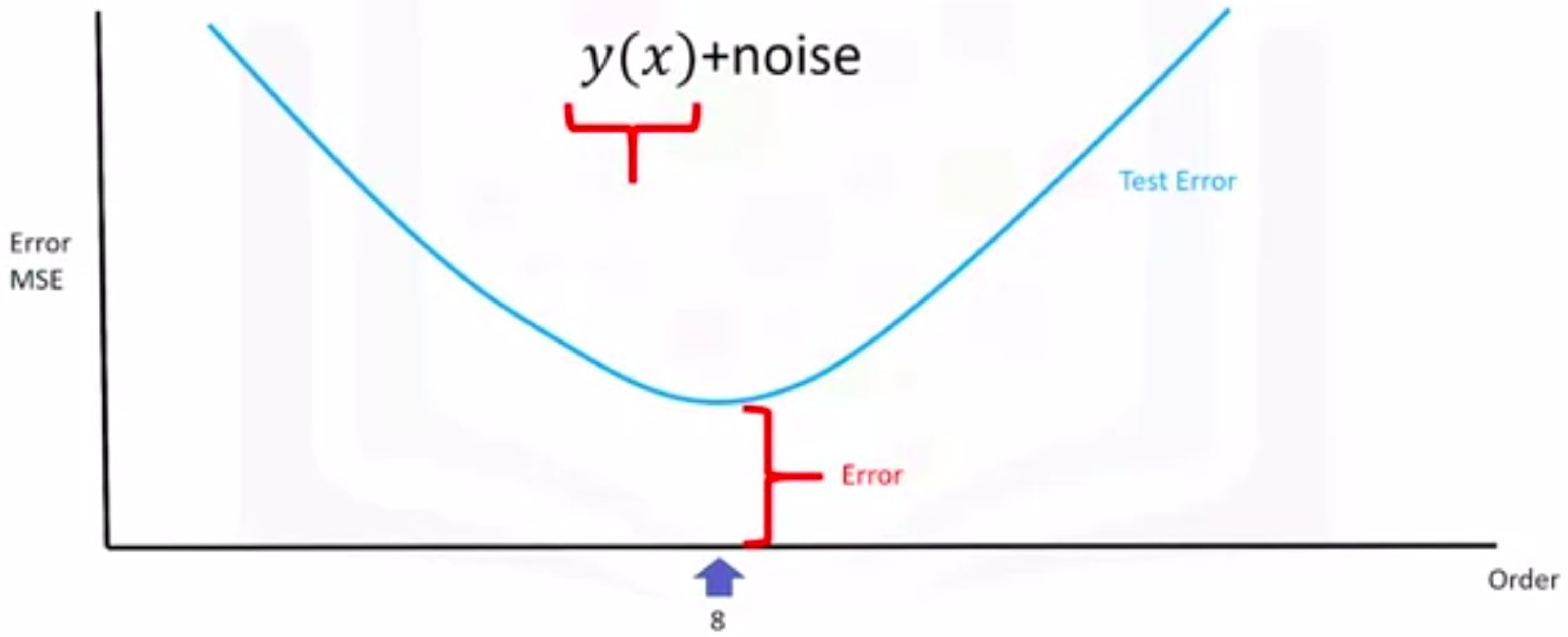
Skip

Continue

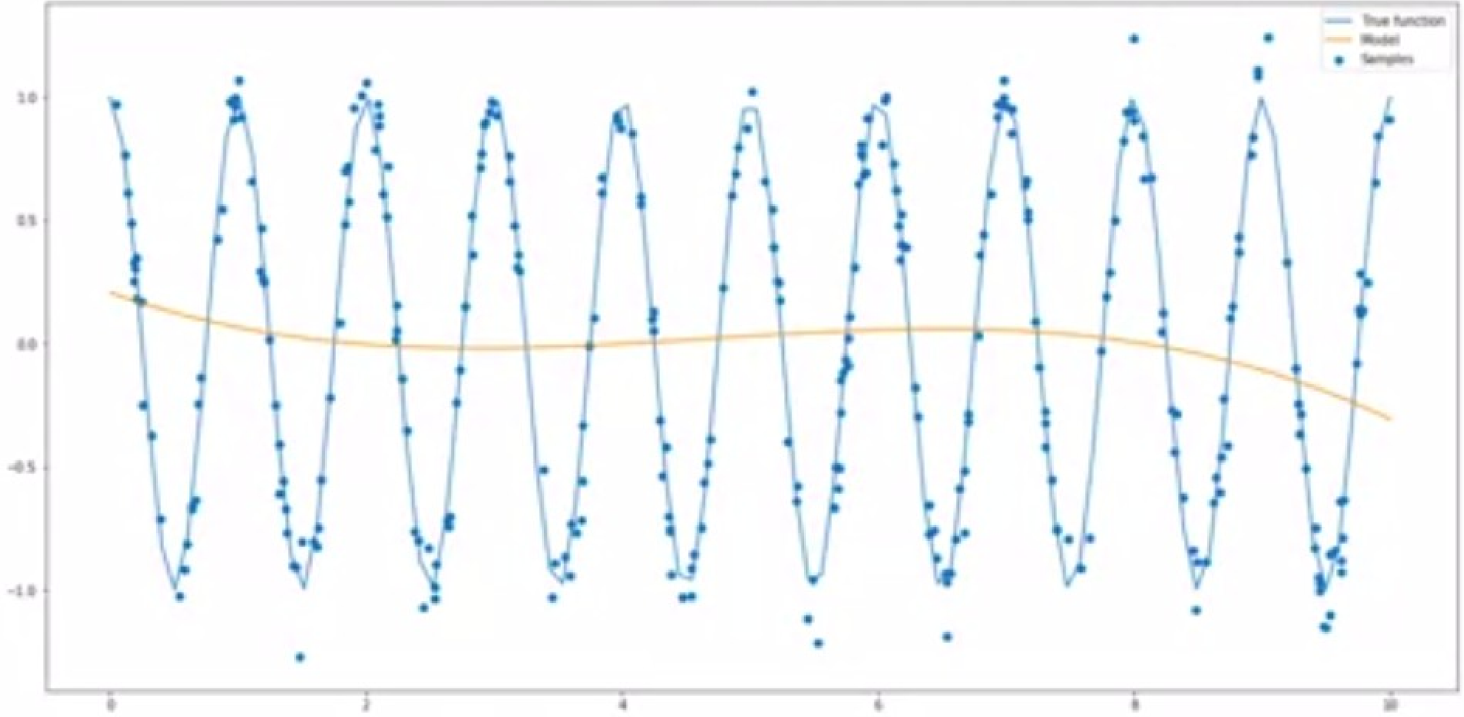
Model Selection



Model Selection

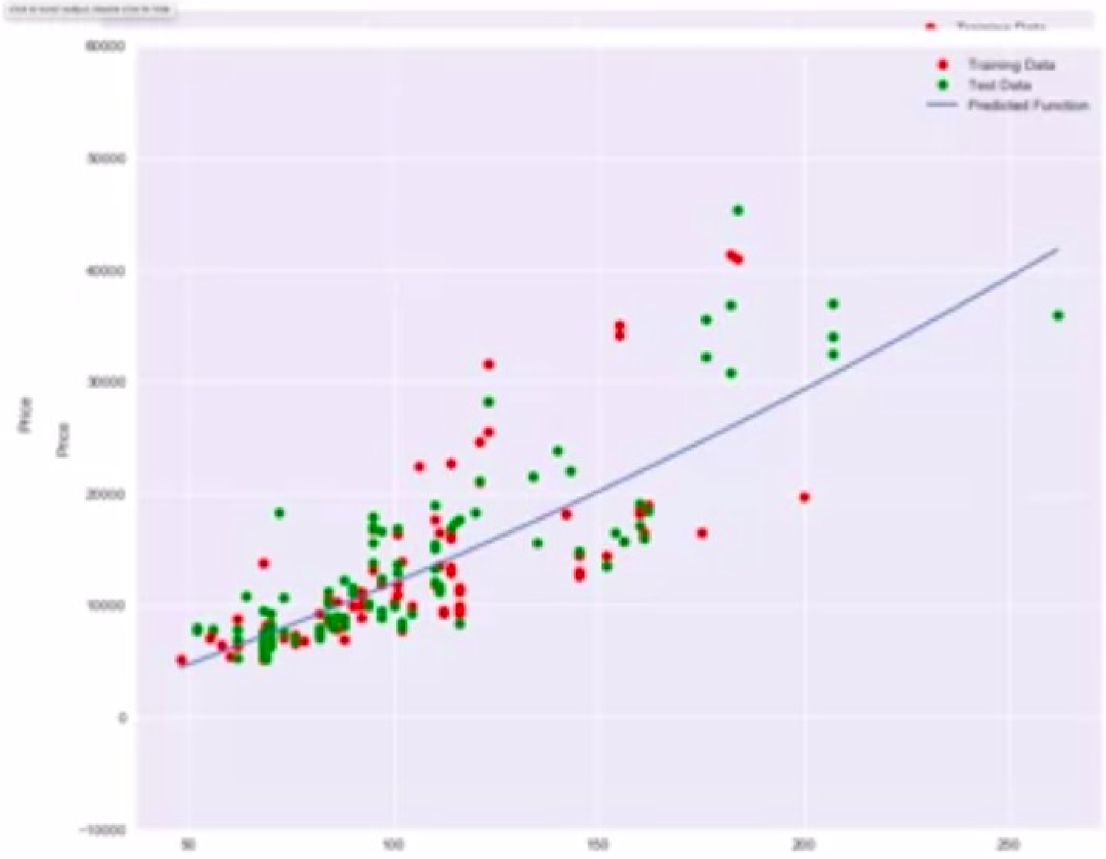


Model Selection




Training Data
Test Data
Predicted Function
50000
40000
30000
20000
10000
0
-10000
Price
50
100
150
200
250

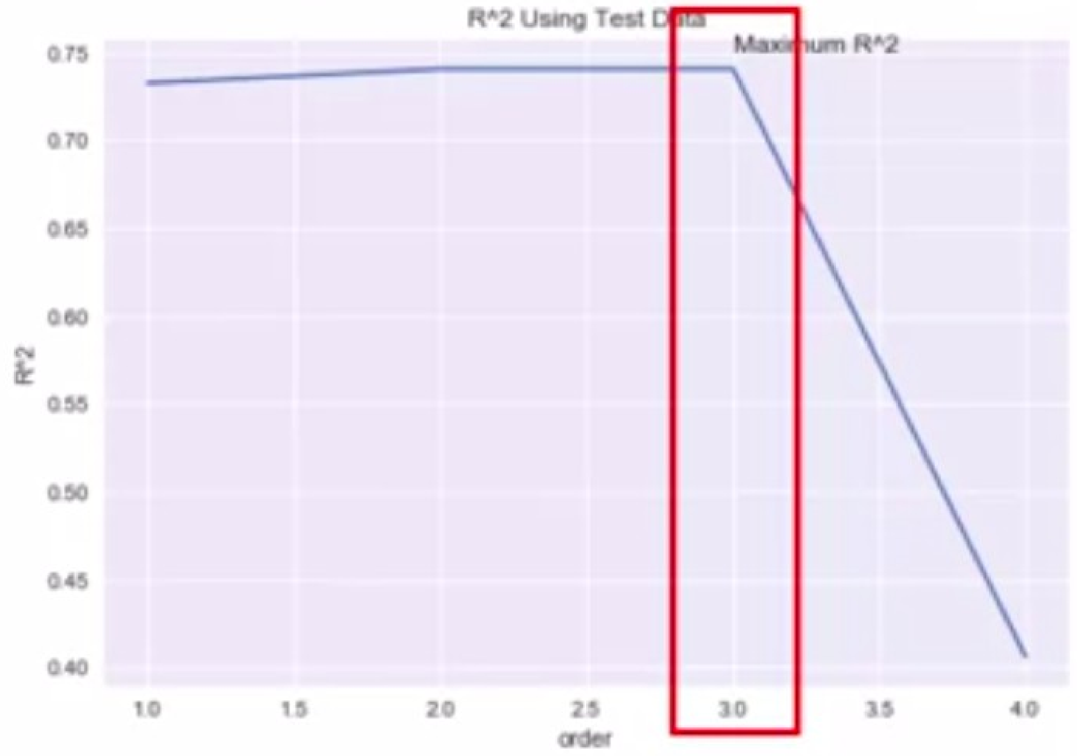

Training Data
Test Data
Predicted Function
Price
50000
40000
30000
20000
10000
0
-10000
50
100
150
200
250


Training Data
Test Data
Predicted Function
Price
60000
50000
40000
30000
20000
10000
0
-10000
50
100
150
200
250

Training Data
Test Data
Predicted Function
Price

Training Data
Test Data
Predicted Function
Price

Model Selection



```
Rsqu_test=[]

order=[1,2,3,4]

for n in order:

    pr=PolynomialFeatures(degree=n)

    x_train_pr=pr.fit_transform(x_train[['horsepower']])

    x_test_pr=pr.fit_transform(x_test[['horsepower']])

    lr.fit(x_train_pr,y_train)

    Rsqu_test.append(lr.score(x_test_pr,y_test))
```

Practice Quiz: Overfitting, Underfitting and Model Selection

Practice Quiz • 3 min • 1 total point

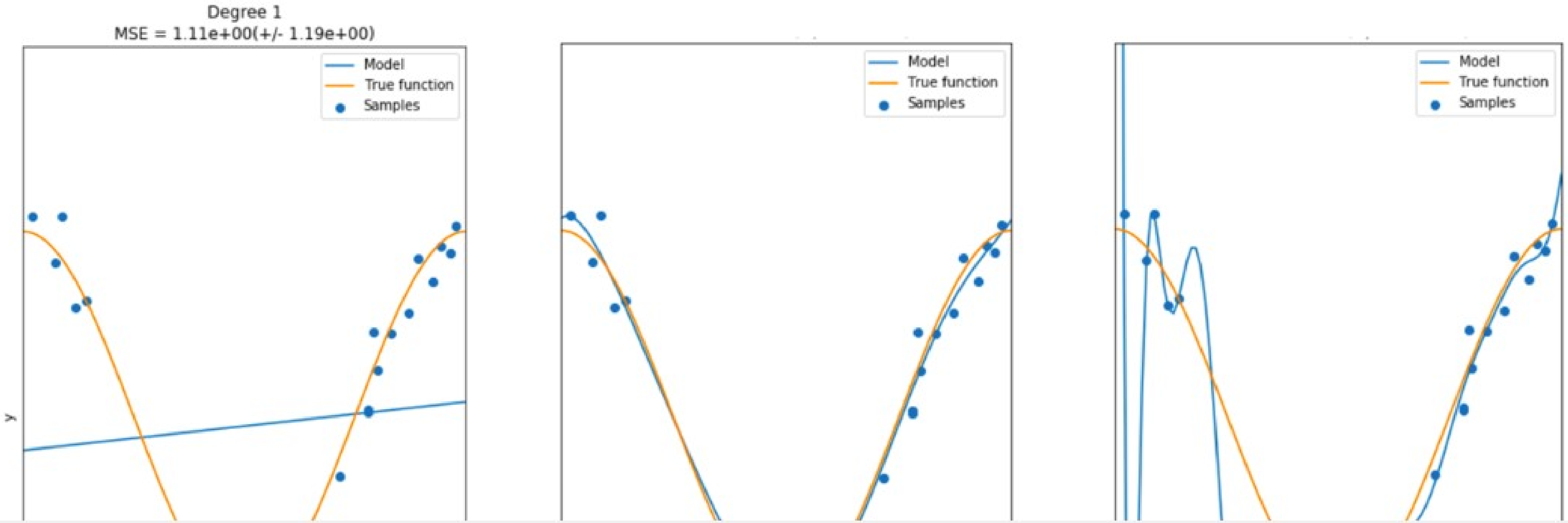
Congratulations! You passed!

Grade received 100% **To pass** 50% or higher

Go to next item

1. what model should you select

1 / 1 point

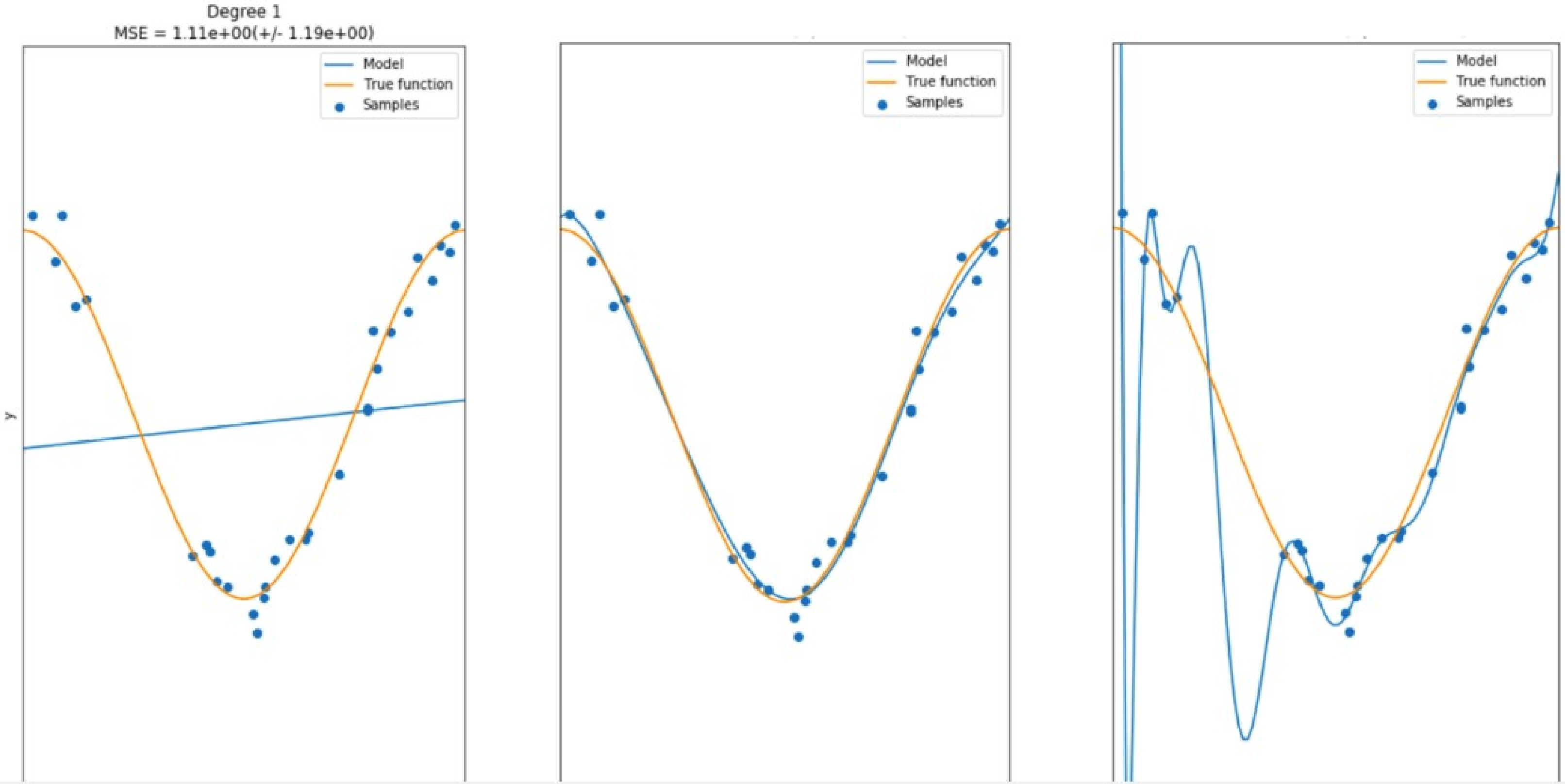


Practice Quiz: Overfitting, Underfitting and Model Selection

Practice Quiz • 3 min • 1 total point

1. what model should you select

1 / 1 point



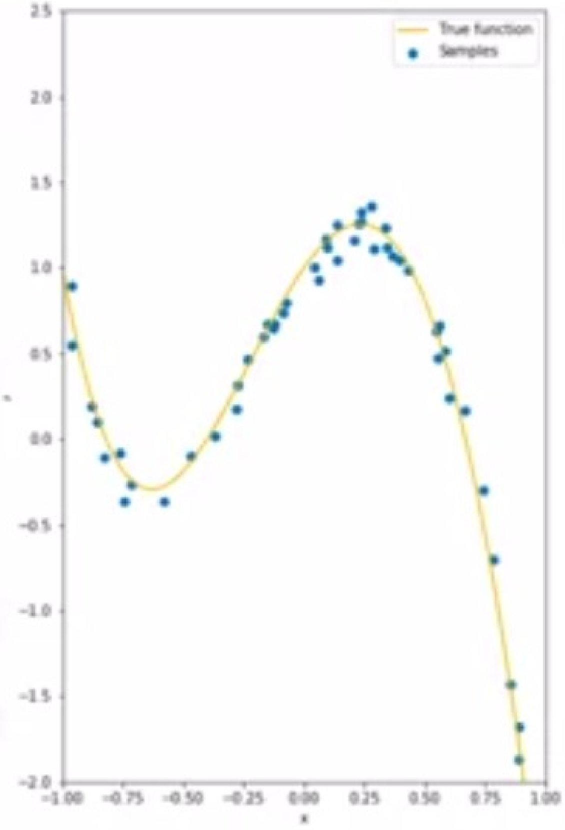
Ridge Regression Introduction

Ridge regression is a regression that is employed in a Multiple regression model when Multicollinearity occurs. Multicollinearity is when there is a strong relationship among the independent variables. Ridge regression is very common with polynomial regression. The next video shows how Ridge regression is used to regularize and reduce the standard errors to avoid over-fitting a regression model

Ridge Regression

Ridge Regression

$$y = 1 + 2x - 3\,x^2 - 4x^3 + x^4$$

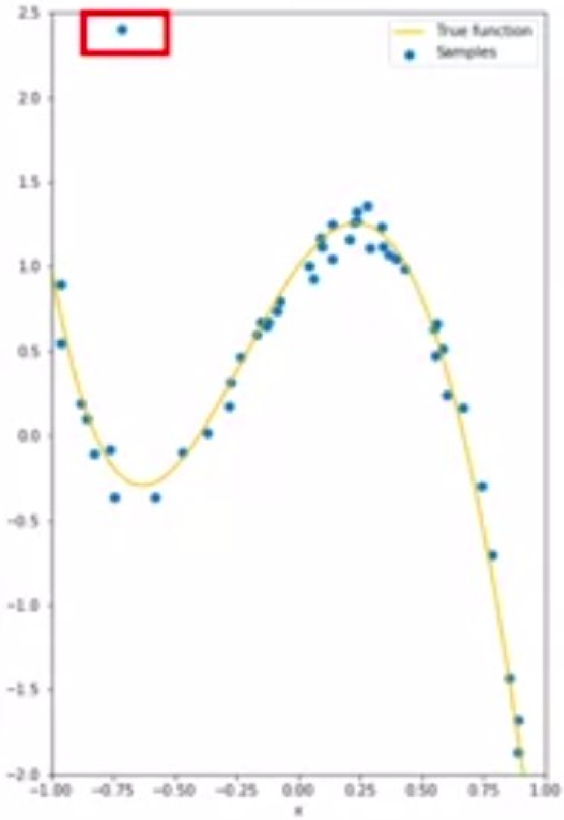


IBM Developer

SKILLS NETWORK

Ridge Regression

$1 + 2x - 3x^2 - 4x^3 + x^4$

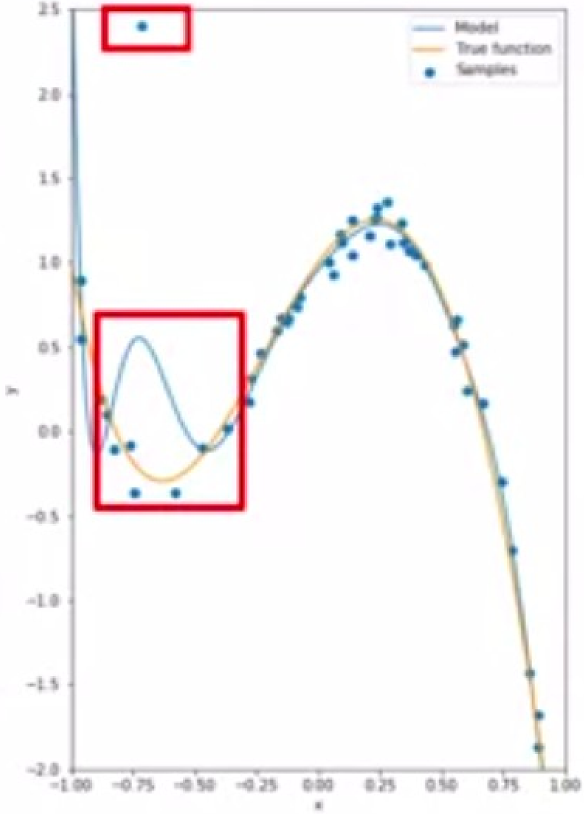


IBM Developer

SKILLS NETWORK

Ridge Regression

$1 + 2x - 3x^2 - 4x^3 + x^4$



IBM Developer

SKILLS NETWORK

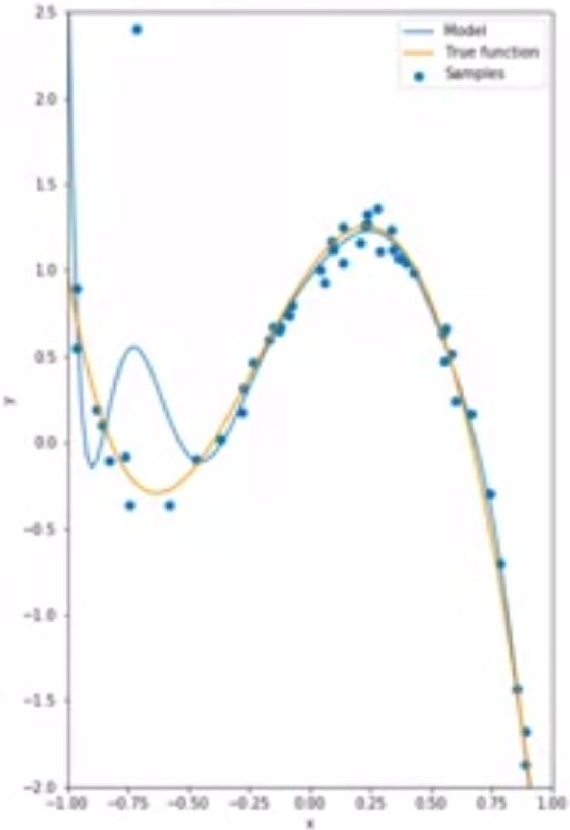
Ridge Regression

$$\hat{y} = 1 + 2\,x - 3\,x^2 - 2x^3 \boxed{- 12\,x^4 - 40x^5 + 80x^6 + 71x^7 - 141x^8 - 38x^9 + 75x^{10}}$$

Alpha	x	x^2	x^3	x^4	x^5	x^6	x^7	x^8	x^9	x^{10}
0	2	-3	-2	-12	-40	80	71	-141	-38	75
0.001	2	-3	-7	5	4	-6	4	-4	4	6
0.01	1	-2	-5	-0.04	0.15	-1	1	-0.5	0.3	1
1	0.5	-1	-1	-0.614	0.70	-0.38	-0. 56	-0.21	-0.5	-0.1
10	0	-0.5	-0.3	-0.37	-0.30	-0.30	-0.22	-0.22	-0.22	-0.17

Ridge Regression

alpha
$\boxed{0}$
0.001
0.01
1
10

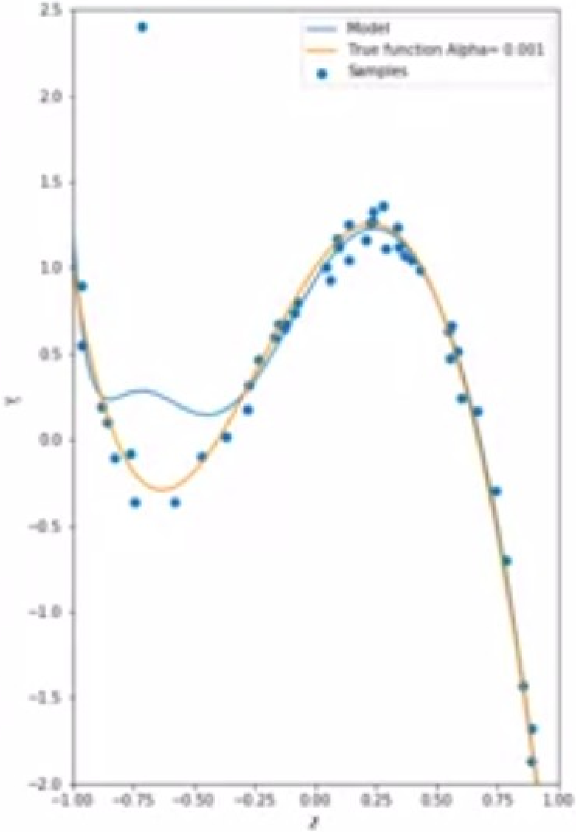


IBM Developer

SKILLS NETWORK

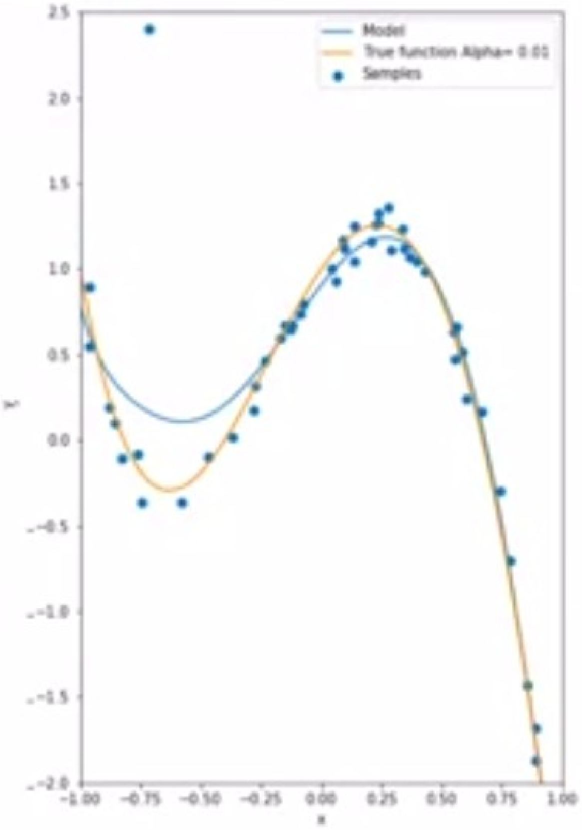
Ridge Regression

alpha
0
0.001
0.01
1
10



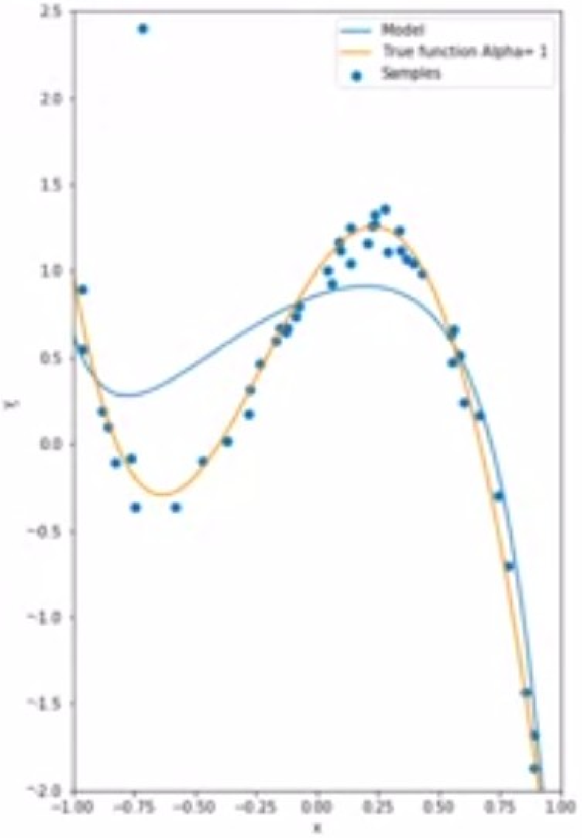
Ridge Regression

alpha
0
0.001
0.01
1
10



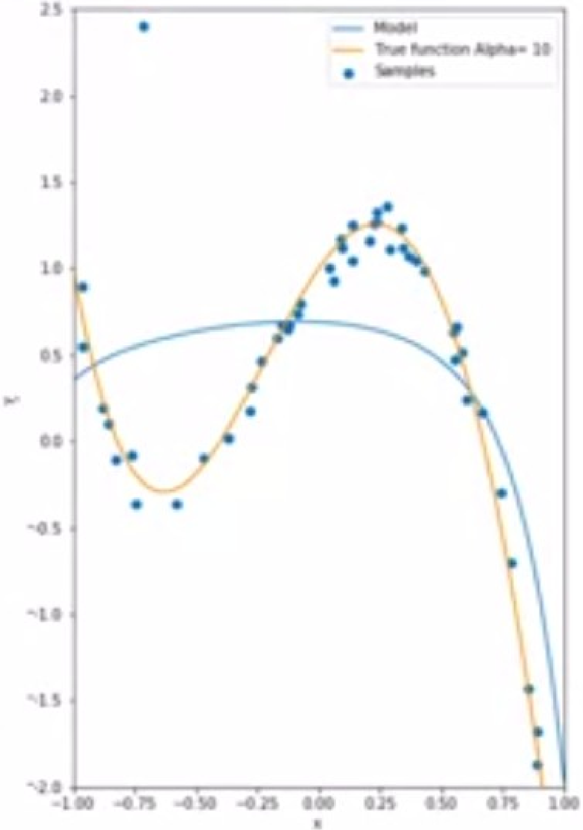
Ridge Regression

alpha
0
0.001
0.01
1
10



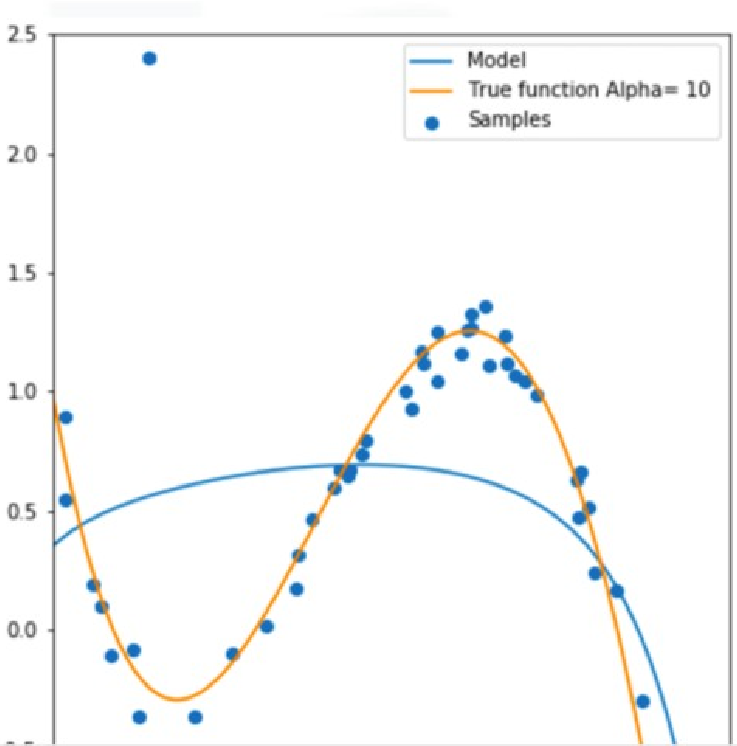
Ridge Regression

alpha
0
0.001
0.01
1
10



Question

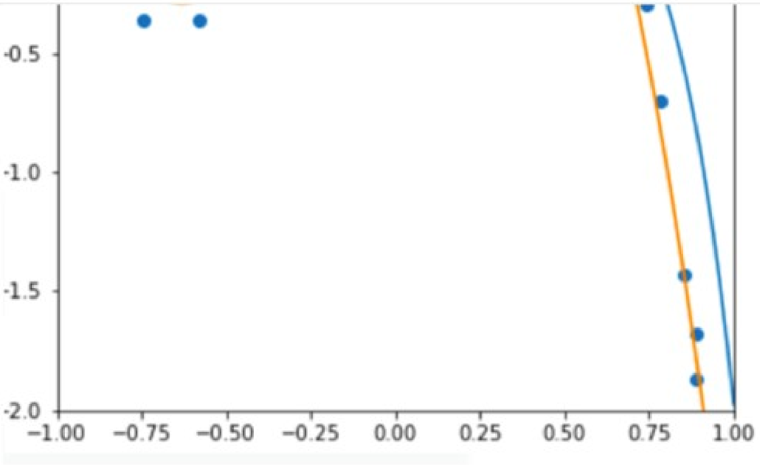
Consider the following fourth order polynomial, fitted with Ridge Regression; should we increase or decrease the parameter alpha?



Skip

Continue

Question



- Decrease
- Increase

Correct

correct

Skip

Continue

Ridge Regression

```
from sklearn.linear_model import Ridge

RidgeModel=Ridge(alpha=0.1)

RidgeModel.fit(X,y)

Yhat=RidgeModel.predict(X)
```

Ridge Regression

Ridge Regression

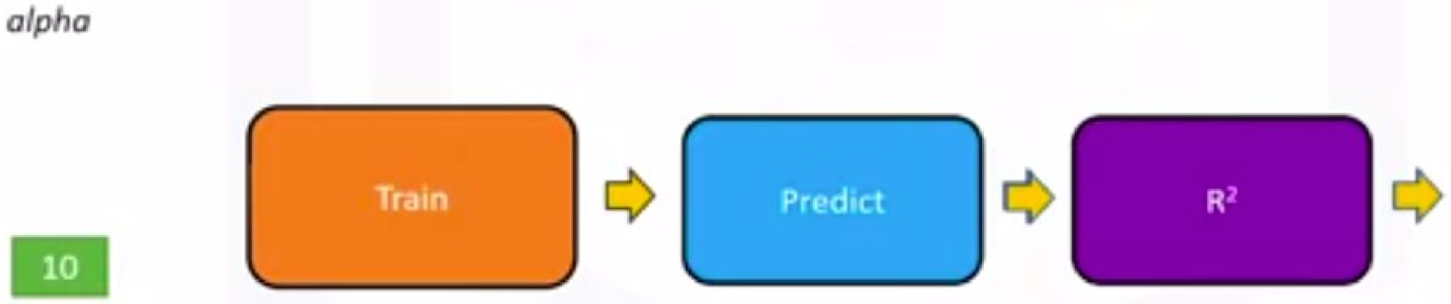
Ridge Regression

Ridge Regression

Ridge Regression

Ridge Regression

Ridge Regression

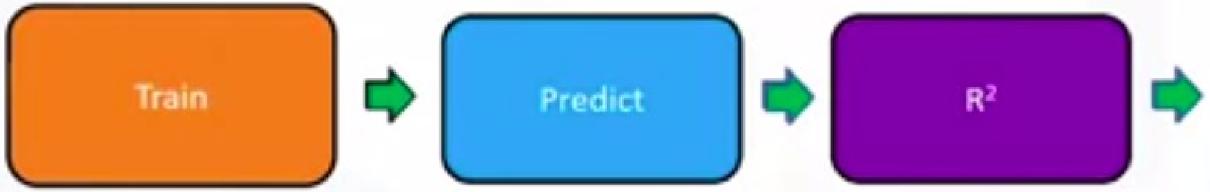


R^2
0.5
0.75

Ridge Regression

Ridge Regression

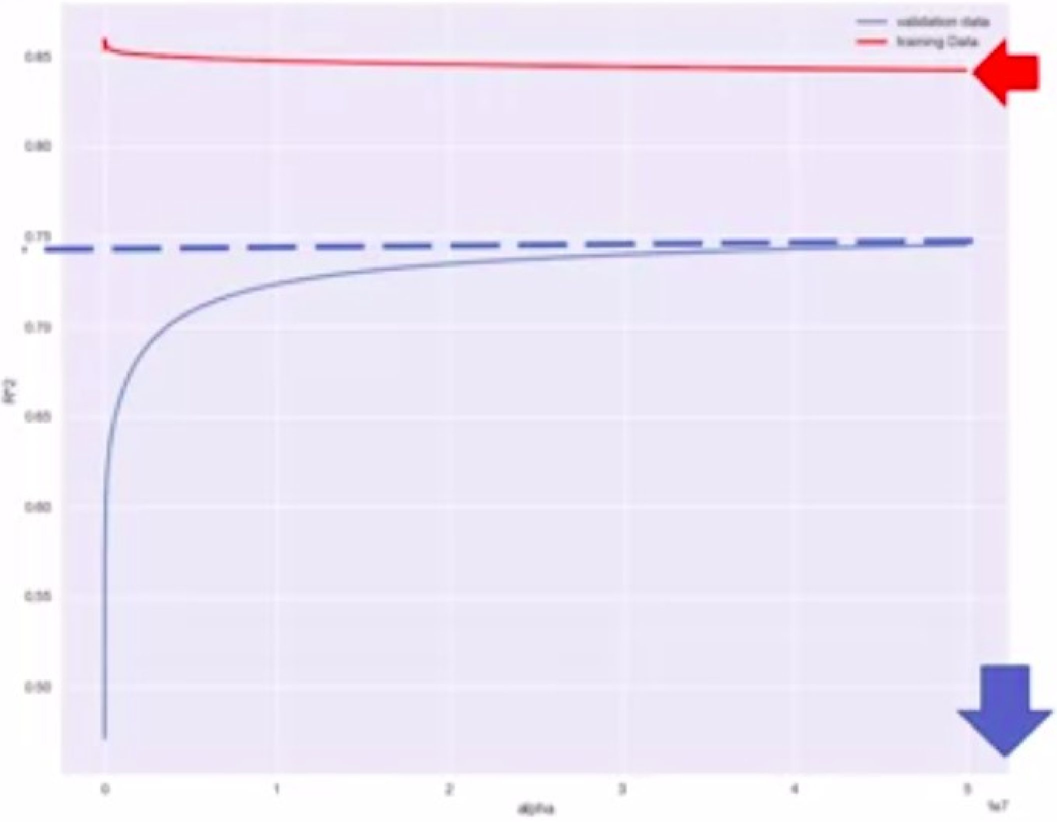
alpha



R^2
0.5
0.75
0.55

Ridge Regression

Ridge Regression

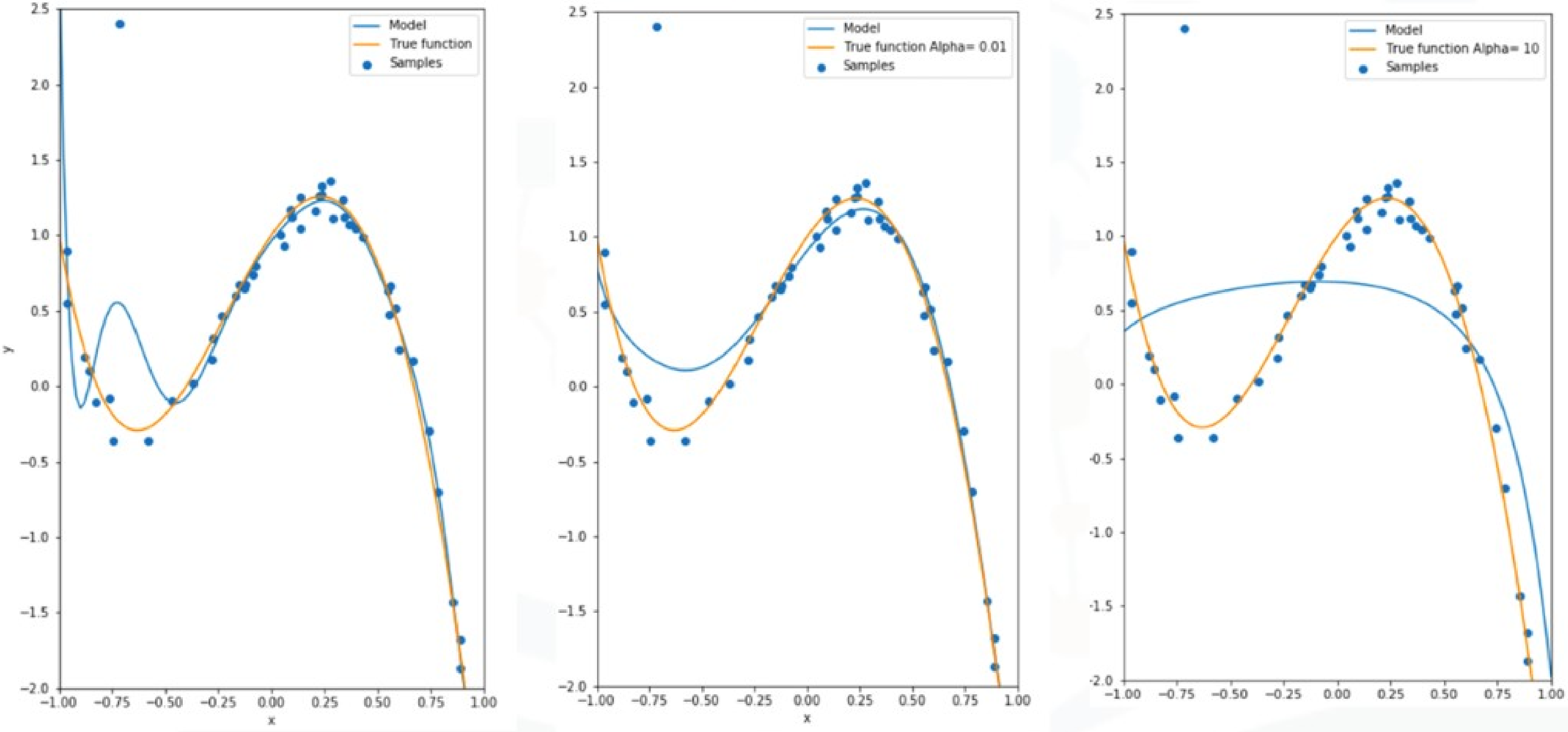


Practice Quiz: Ridge Regression

Practice Quiz • 3 min • 1 total point

1. the following models were all trained on the same data, select the model with the lowest value for alpha:

1 / 1 point

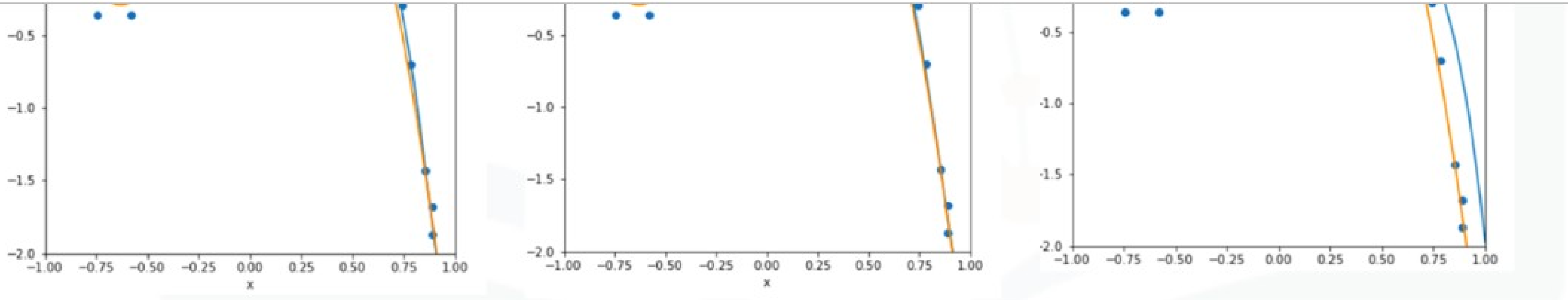


a b c

← Back

Practice Quiz: Ridge Regression

Practice Quiz • 3 min • 1 total point



a **b** **c**

- ◉ a
- ○ b
- ○ c

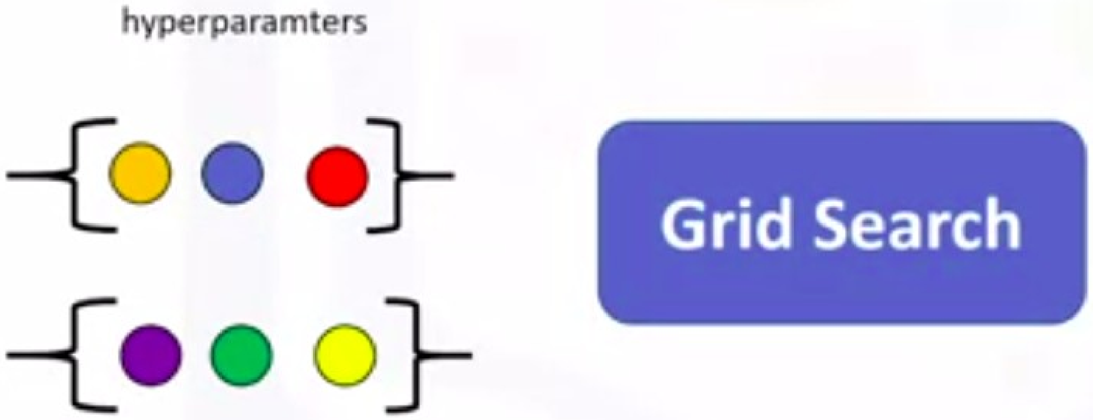
Correct

correct

Grid Search

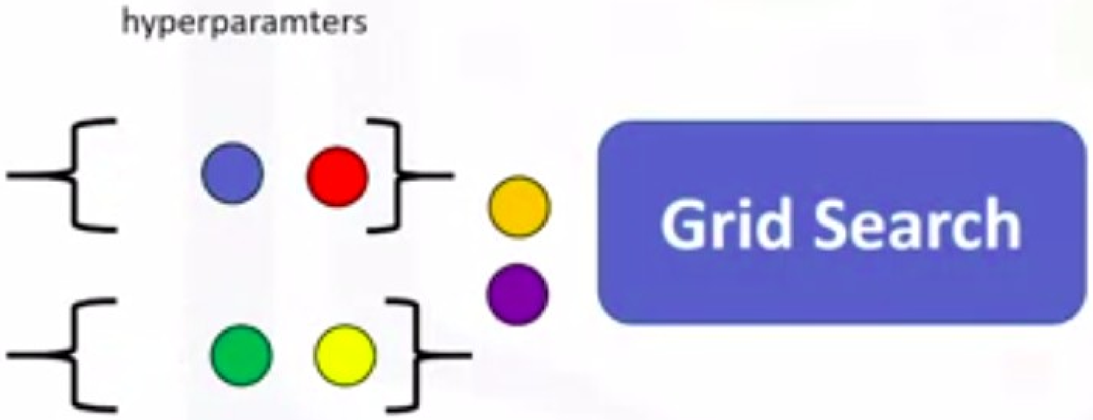
Hyperparameters

- In the last section, the term alpha in Ridge regression is called a hyperparameter
- Scikit-lean has a means of automatically iterating over these hyperparamters using cross-validation called Grid Search



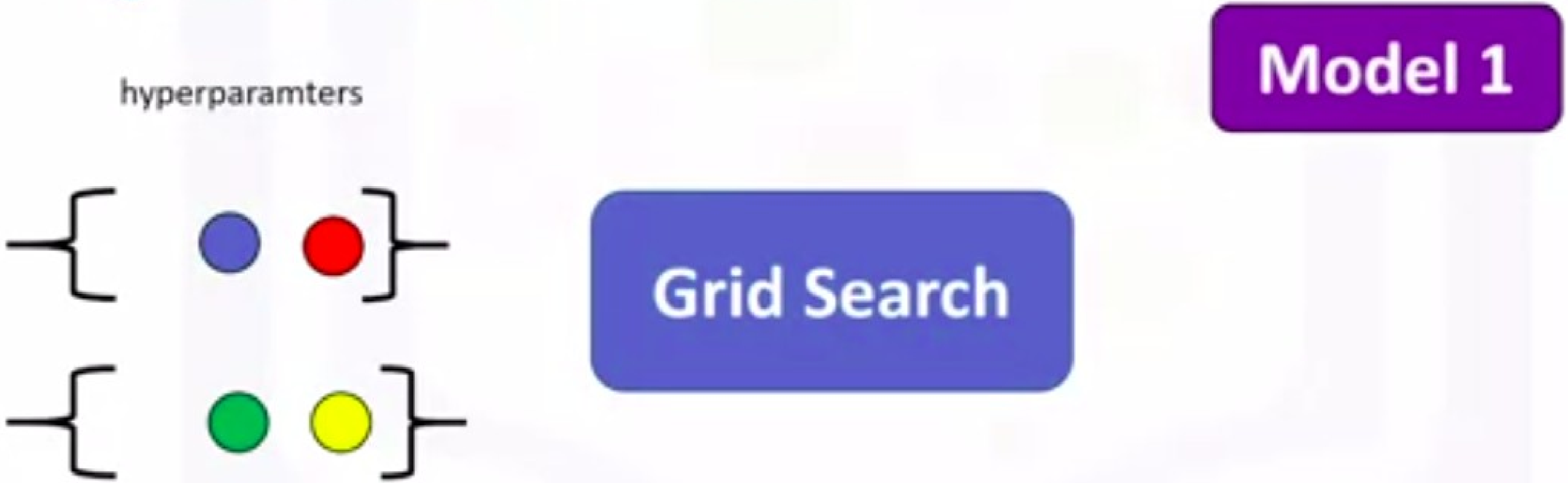
Hyperparameters

- In the last section, the term alpha in Ridge regression is called a hyperparameter
- Scikit-lean has a means of automatically iterating over these hyperparamters using cross-validation called Grid Search



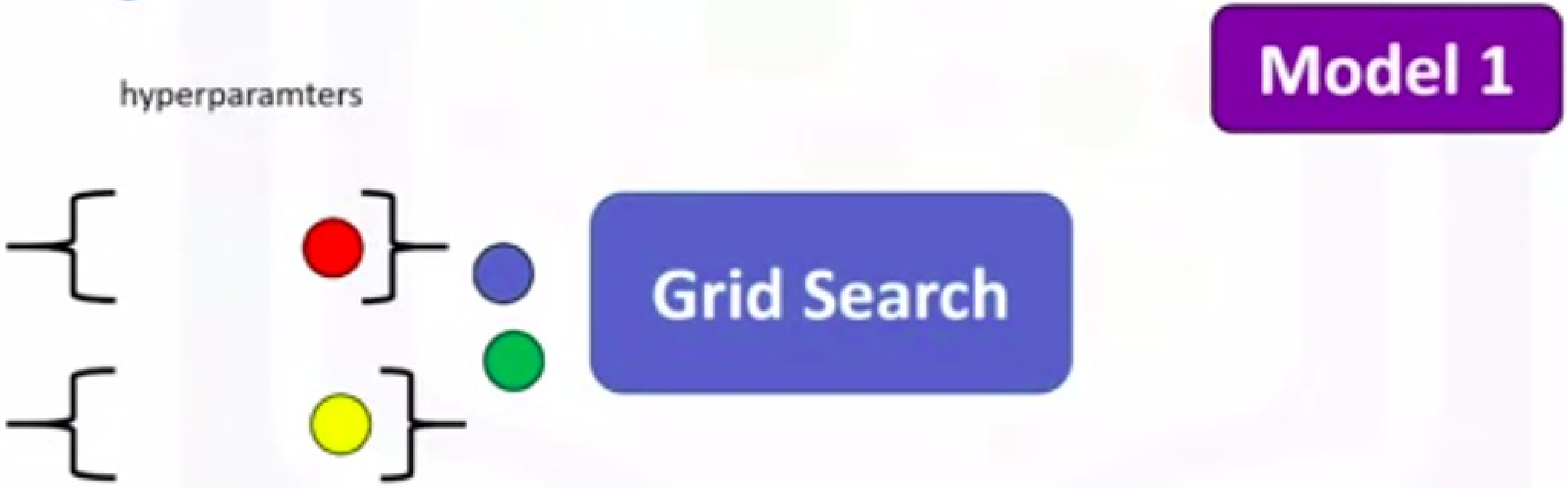
Hyperparameters

- In the last section, the term alpha in Ridge regression is called a hyperparameter
- Scikit-lean has a means of automatically iterating over these hyperparamters using cross-validation called Grid Search



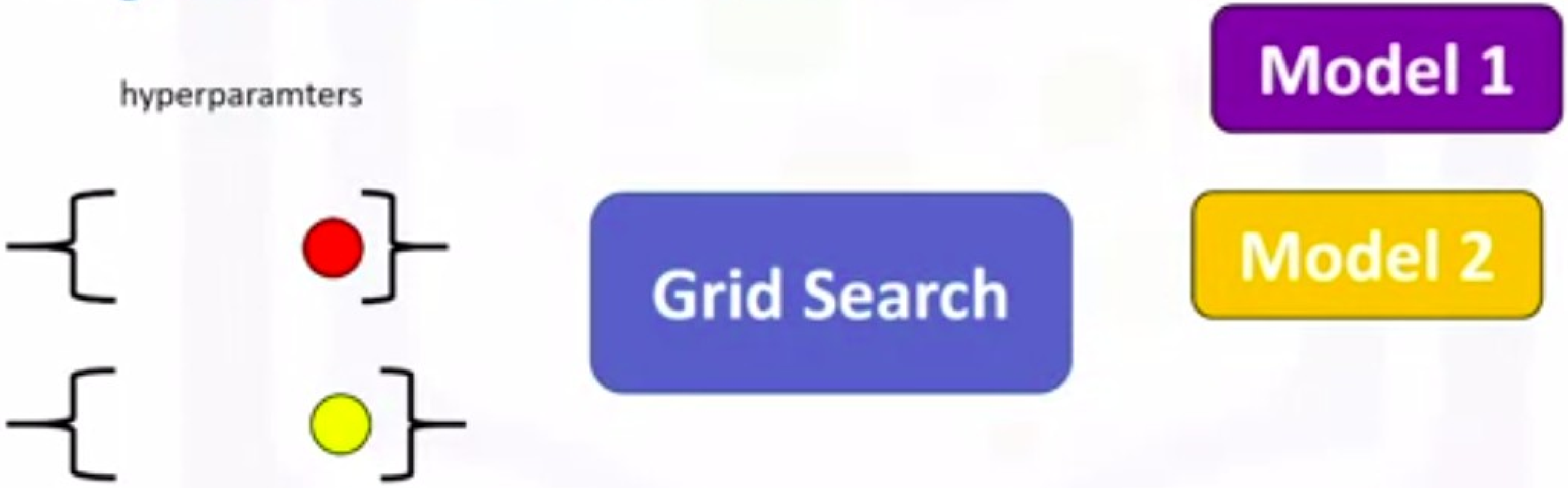
Hyperparameters

- In the last section, the term alpha in Ridge regression is called a hyperparameter
- Scikit-lean has a means of automatically iterating over these hyperparamters using cross-validation called Grid Search



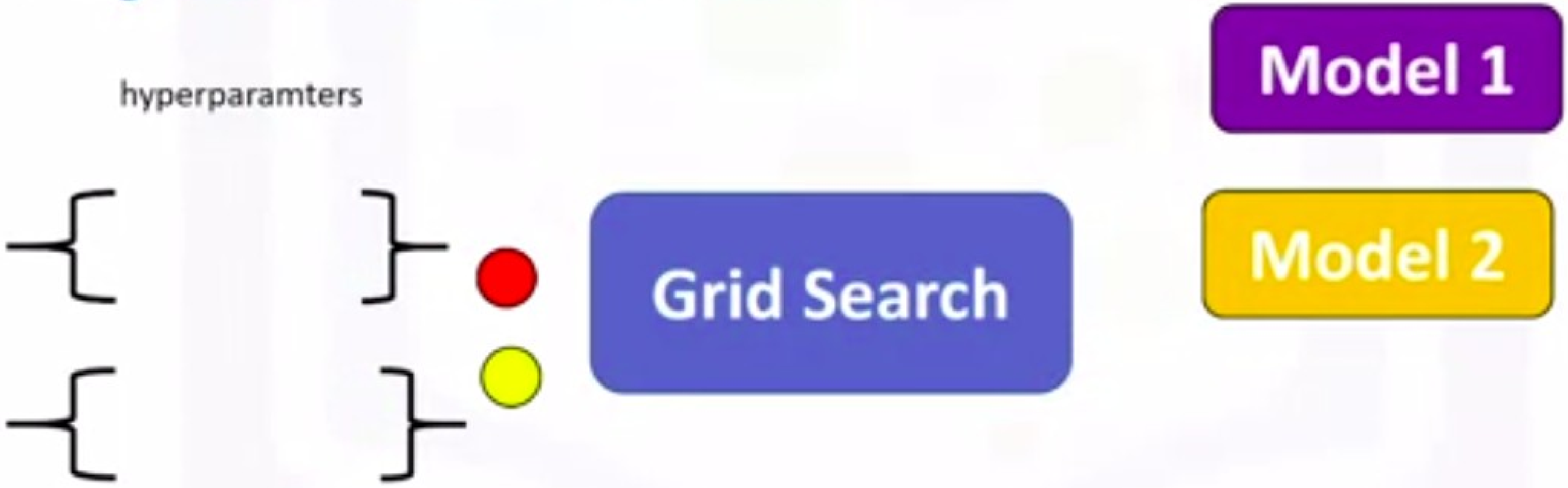
Hyperparameters

- In the last section, the term alpha in Ridge regression is called a hyperparameter
- Scikit-lean has a means of automatically iterating over these hyperparamters using cross-validation called Grid Search



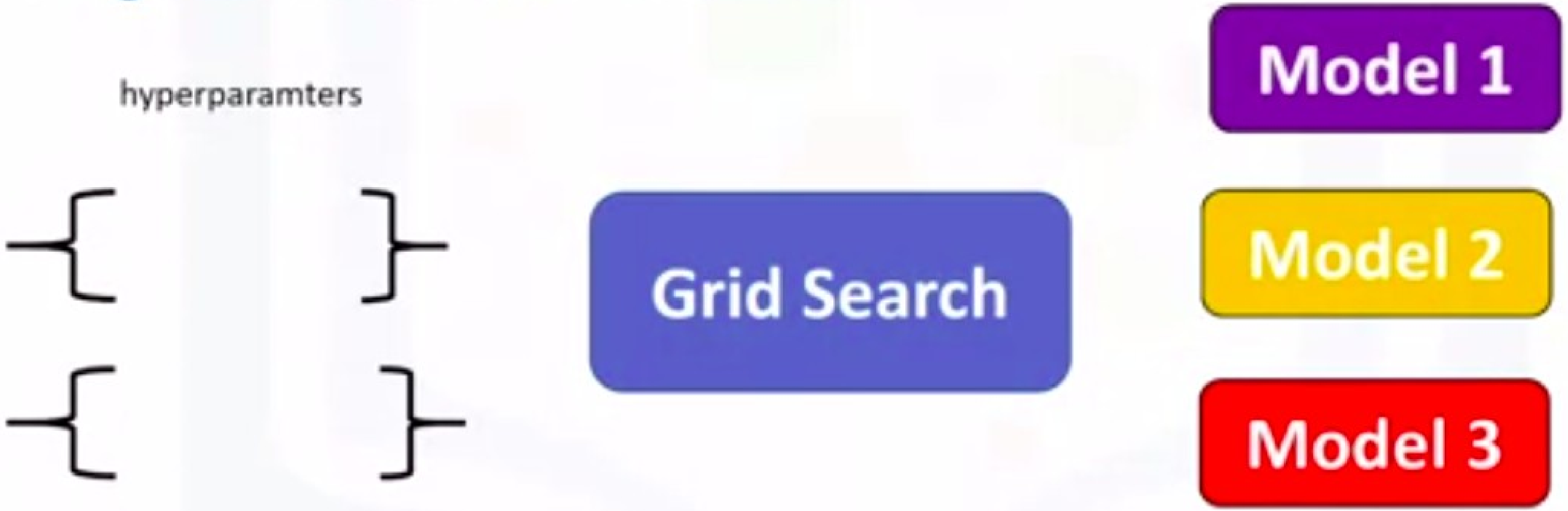
Hyperparameters

- In the last section, the term alpha in Ridge regression is called a hyperparameter
- Scikit-lean has a means of automatically iterating over these hyperparamters using cross-validation called Grid Search



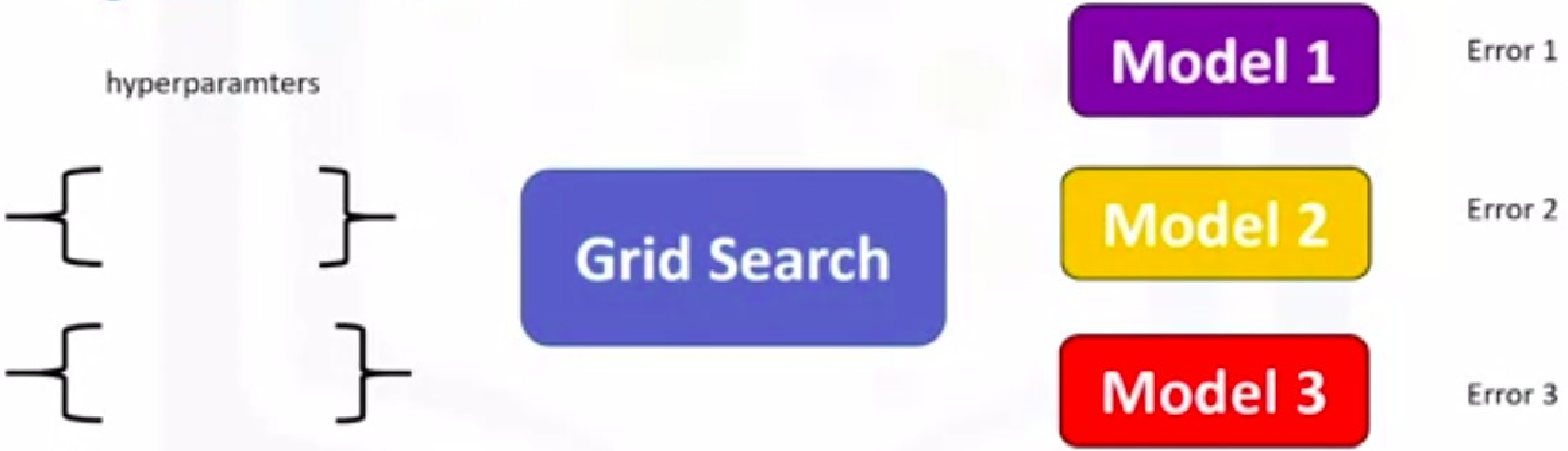
Hyperparameters

- In the last section, the term alpha in Ridge regression is called a hyperparameter
- Scikit-lean has a means of automatically iterating over these hyperparamters using cross-validation called Grid Search



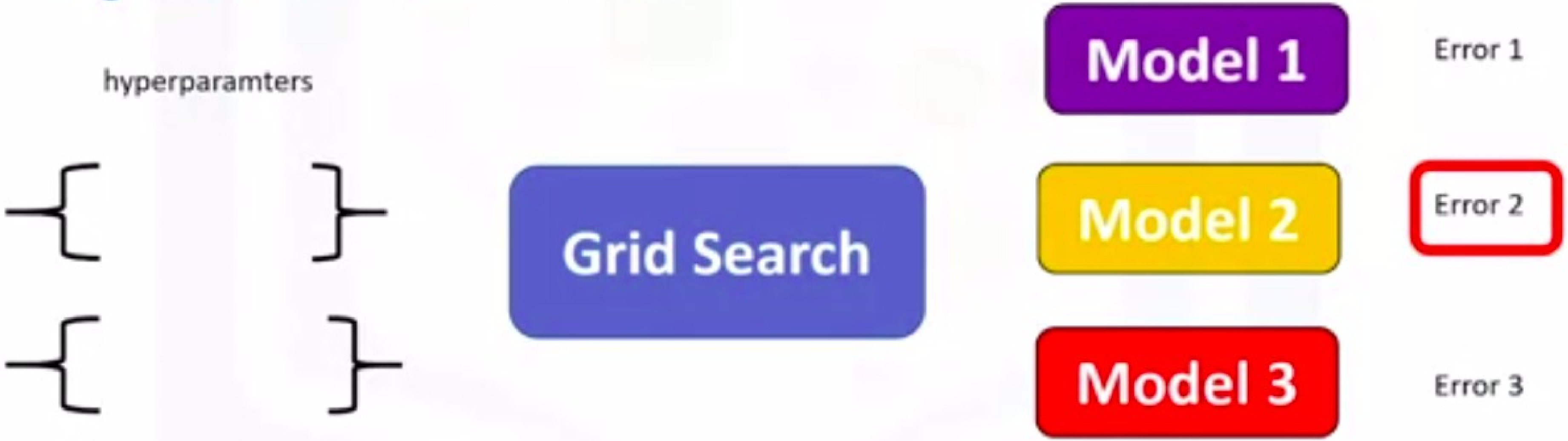
Hyperparameters

- In the last section, the term alpha in Ridge regression is called a hyperparameter
- Scikit-lean has a means of automatically iterating over these hyperparamters using cross-validation called Grid Search



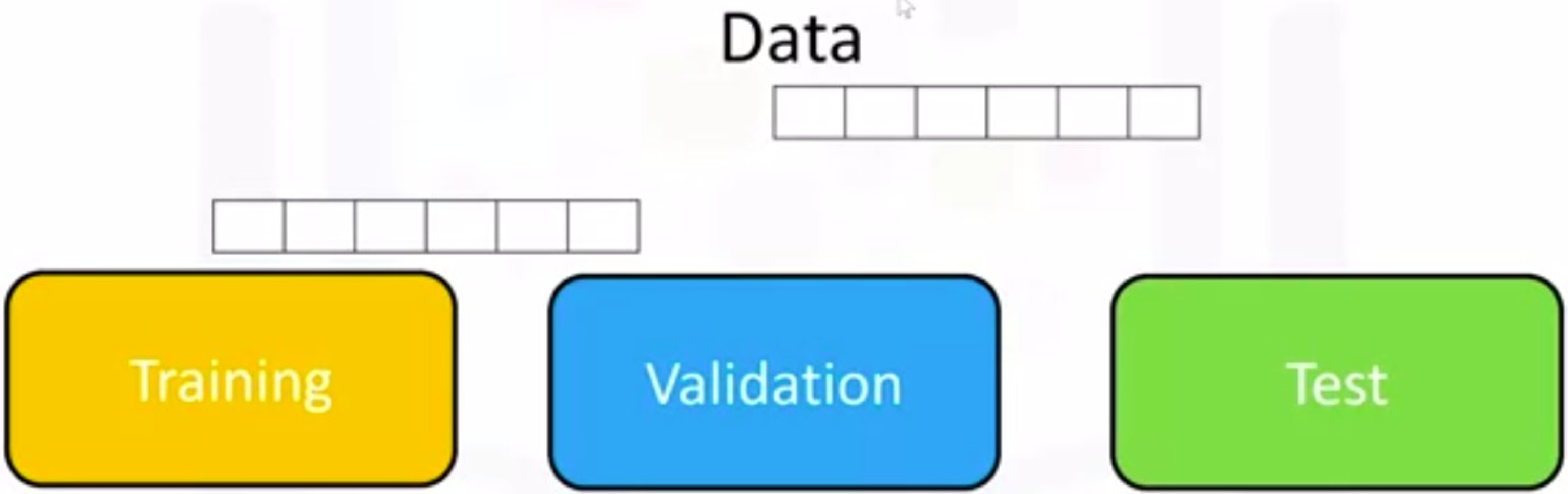
Hyperparameters

- In the last section, the term alpha in Ridge regression is called a hyperparameter
- Scikit-lean has a means of automatically iterating over these hyperparamters using cross-validation called Grid Search

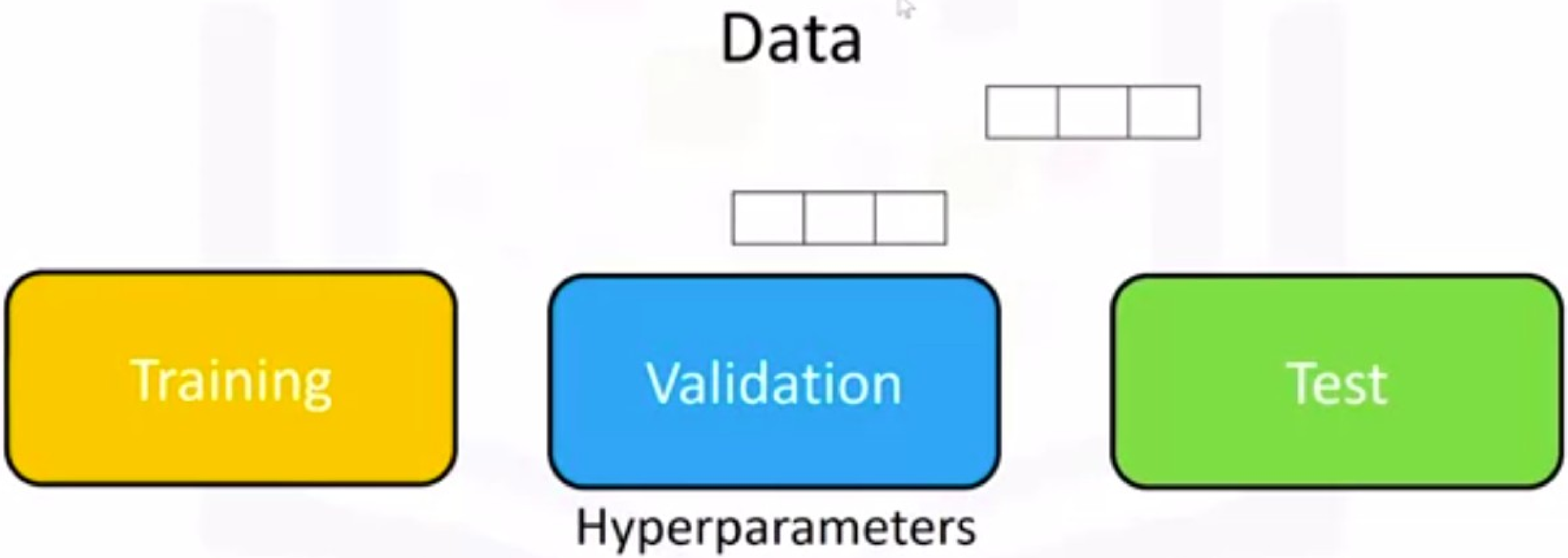


Grid Search

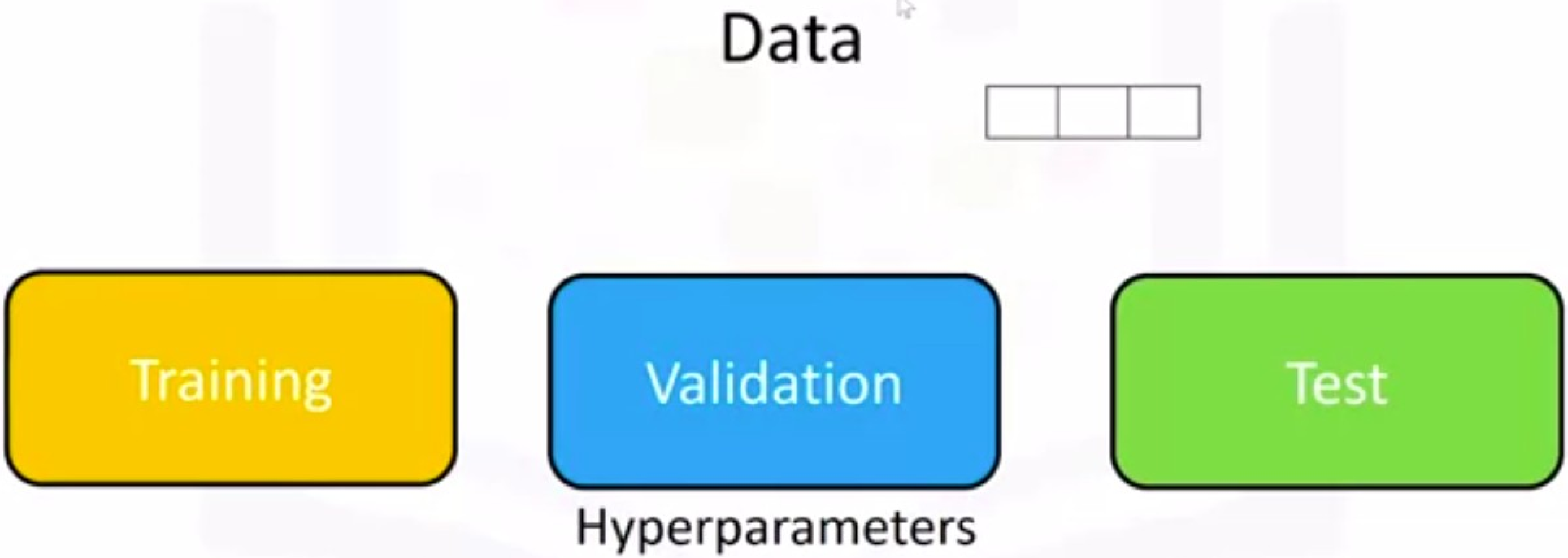
Grid Search



Grid Search



Grid Search



Question

What data do we use to pick the best hyperparameter

- ◯ Training data
- ◯ Validation data
- ◯ Test data

Skip Submit

IBM Developer SKILLS NETWORK

Question

What data do we use to pick the best hyperparameter

- ○ Training data
- ◉ Validation data
- ○ Test data

Correct
correct

Skip

Continue

Home Installation Documentation Examples

Google Custom Search Search

Previous Next Up

scikit-learn v0.19.0

Other versions

Please cite us if you use the software.

sklearn.linear_model.Ridge

Examples using sklearn.linear_model.Ridge

sklearn.linear_model.Ridge

class sklearn.linear_model.Ridge(alpha=1.0, fit_intercept=True, normalize=False, copy_X=True, max_iter=None, tol=0.001, solver='auto', random_state=None) [source]

Linear least squares with l2 regularization.

This model solves a regression model where the loss function is the linear least squares function and regularization is given by the l2-norm. Also known as Ridge Regression or Tikhonov regularization. This estimator has built-in support for multi-variate regression (i.e., when y is a 2d-array of shape [n_samples, n_targets]).

Read more in the User Guide.

Parameters: **alpha** : {float, array-like}, shape (n_targets)

Regularization strength; must be a positive float. Regularization improves the conditioning of the problem and reduces the variance of the estimates. Larger values specify stronger regularization. Alpha corresponds to `C^-1` in other linear models such as LogisticRegression or LinearSVC. If an array is passed, penalties are assumed to be specific to the targets. Hence they must correspond in number.

fit_intercept : boolean

Whether to calculate the intercept for this model. If set to false, no intercept will be used in calculations (e.g. data is expected to be already centered).

normalize : boolean, optional, default False

This parameter is ignored when `fit_intercept` is set to False. If True, the regressors X will be normalized before regression by subtracting the mean and dividing by the l2-norm. If you wish to standardize, please use `sklearn.preprocessing.StandardScaler` before calling `fit` on an estimator with `normalize=False`.

copy_X : boolean, optional, default True

If True, X will be copied; else, it may be overwritten.

Grid Search

parameters = [{ 'alpha' : [1, 10, 100, 1000] }]

Alpha	1	10	100	1000

Ridge()

Grid Search

Ridge()

Scoring

Number of Folds

Grid Search CV

Alpha	1	10	100	1000

Grid Search

Ridge()

Number of Folds

Grid Search CV

Alpha	1	10	100	1000

Grid Search

Alpha	1	10	100	1000

Grid Search

Grid Search CV

Alpha	1	10	100	1000

Grid Search

Grid Search CV

Alpha	1	10	100	1000
R^2	0.74	0.35	0.073	0.008

Grid Search

Grid Search CV

Alpha	1	10	100	1000
R^2	0.74	0.35	0.073	0.008

```python
from sklearn.linear_model import Ridge
from sklearn.model_selection import GridSearchCV

parameters1= [{'alpha': [0.001,0.1,1, 10, 100, 1000,10000,100000,100000]}]

RR=Ridge()

Grid1 = GridSearchCV(RR, parameters1,cv=4)

Grid1.fit(x_data[['horsepower', 'curb-weight', 'engine-size', 'highway-mpg']],y_data)

Grid1.best_estimator_

scores  = Grid1.cv_results_
scores['mean_test_score']
```

```
array([ 0.66549413, 0.66554568, 0.66602936, 0.66896822, 0.67334636, 0.65781884, 0.65781884])
```

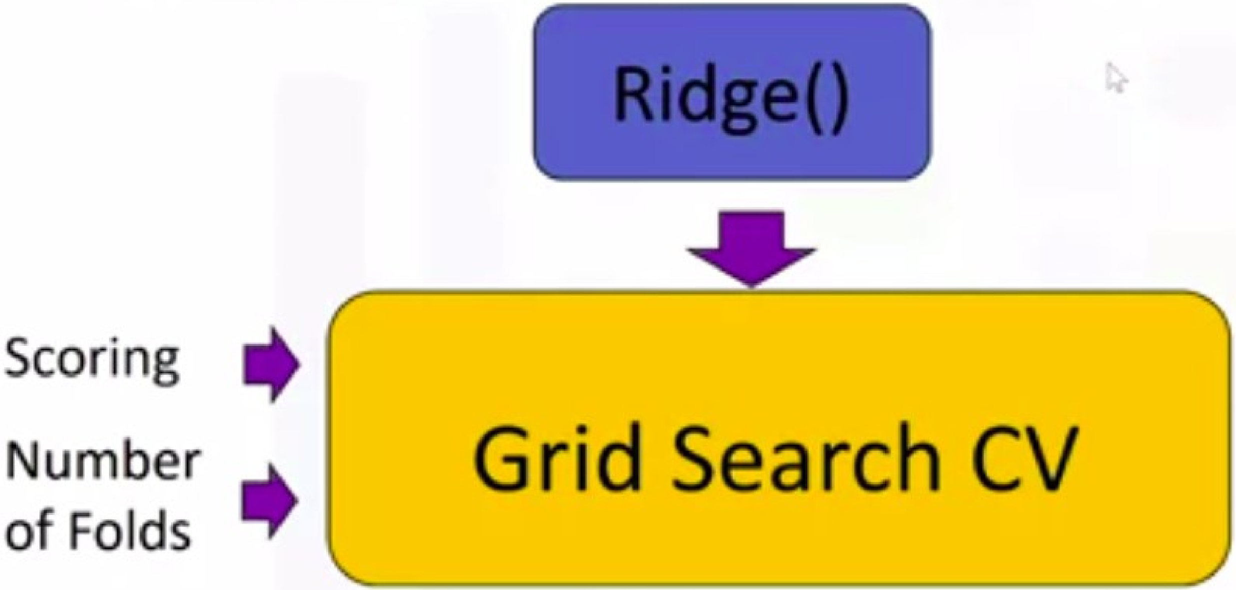
Grid Search

```
parameters = [ { 'alpha' : [1, 10, 100, 1000], 'normalize' : [True, False] } ]
```

Alpha	1	10	100	1000
Normalize	**True**	**True**	**True**	**True**
	False	**False**	**False**	**False**

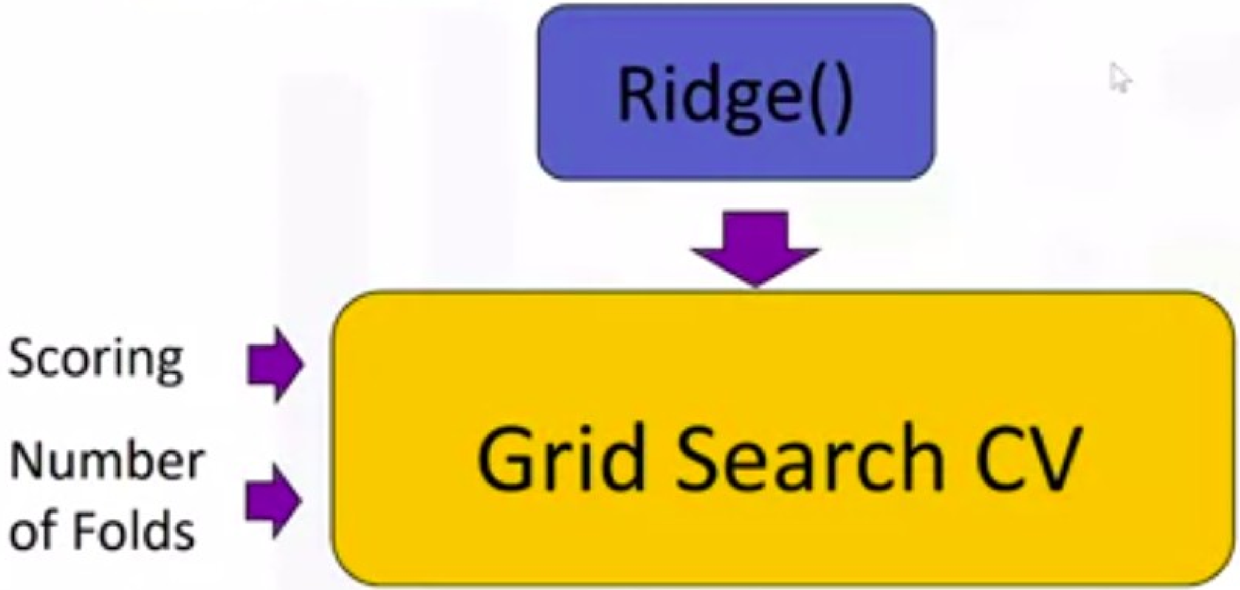
Ridge()

Grid Search



Alpha	1	10	100	1000
Normalize	True	True	True	True
	False	False	False	False

Grid Search



Alpha	1	10	100	1000
True	0.69	0.32	0.17	0.17
False	0.67	0.66	0.66	0.64

```python
from sklearn.linear_model import Ridge
from sklearn.model_selection import GridSearchCV

parameters2= [{'alpha': [0.001,0.1,1, 10, 100], 'normalize' :  [True,  False] }]

RR=Ridge()

Grid1 = GridSearchCV(RR, parameters2,cv=4)

Grid1.fit(x_data[['horsepower', 'curb-weight', 'engine-size', 'highway-mpg']],y_data)

Grid1.best_estimator_

scores  = Grid1.cv_results_
```

```
for param,mean_val, mean_test  inzip(scores['params'],scores['mean_test_score'],scores['mean_train_score']):

        print(param, "R^2 on test data:", mean_val,"R^2 on train data:" ,mean_test)
```

```
{'alpha': 0.001, 'normalize': True} R^2 on tesst data: 0.66605547293 R^2 on train data: 0.814001968709
{'alpha': 0.001, 'normalize': False} R^2 on tesst data: 0.665488366584 R^2 on train data: 0.814002698797
{'alpha': 0.1, 'normalize': True} R^2 on tesst data: 0.694175625356 R^2 on train data: 0.810546768311
{'alpha': 0.1, 'normalize': False} R^2 on tesst data: 0.665488937796 R^2 on train data: 0.814002698794
{'alpha': 1, 'normalize': True} R^2 on tesst data: 0.690486934584 R^2 on train data: 0.749104440368
{'alpha': 1, 'normalize': False} R^2 on tesst data: 0.665494127178 R^2 on train data: 0.814002698472
{'alpha': 10, 'normalize': True} R^2 on tesst data: 0.321376875232 R^2 on train data: 0.341856042902
{'alpha': 10, 'normalize': False} R^2 on tesst data: 0.665545680812 R^2 on train data: 0.8140026666
{'alpha': 100, 'normalize': True} R^2 on tesst data: 0.0170551710263 R^2 on train data: 0.0496044796826
{'alpha': 100, 'normalize': False} R^2 on tesst data: 0.666029359996 R^2 on train data: 0.813999791851
{'alpha': 1000, 'normalize': True} R^2 on tesst data: -0.0301961745066 R^2 on train data: 0.005184451599
{'alpha': 1000, 'normalize': False} R^2 on tesst data: 0.668968215369 R^2 on train data: 0.813870488264
{'alpha': 10000, 'normalize': True} R^2 on tesst data: -0.0351687400461 R^2 on train data: 0.000520784757979
{'alpha': 10000, 'normalize': False} R^2 on tesst data: 0.673346359342 R^2 on train data: 0.812583743226
{'alpha': 100000, 'normalize': True} R^2 on tesst data: -0.0356685844558 R^2 on train data: 5.2101975528e-05
{'alpha': 100000, 'normalize': False} R^2 on tesst data: 0.657818838432 R^2 on train data: 0.789541446486
{'alpha': 100000, 'normalize': True} R^2 on tesst data: -0.0356685844558 R^2 on train data: 5.2101975528e-05
{'alpha': 100000, 'normalize': False} R^2 on tesst data: 0.657818838432 R^2 on train data: 0.789541446486
```

Question

how many types of parameters does the following dictionary contain:

```
parameters= [{'alpha': [0.001,0.1,1, 10, 100], 'normalize' : [True, False] }]
```

- (selected) 2
- 4
- 9

Correct

correct

Skip

Continue

Lesson Summary

In this lesson, you have learned how to:

Identify over-fitting and under-fitting in a predictive model: Overfitting occurs when a function is too closely fit to the training data points and captures the noise of the data. Underfitting refers to a model that can't model the training data or capture the trend of the data.

Apply Ridge Regression to linear regression models: Ridge regression is a regression that is employed in a Multiple regression model when Multicollinearity occurs.

Tune hyper-parameters of an estimator using Grid search: Grid search is a time-efficient tuning technique that exhaustively computes the optimum values of hyperparameters performed on specific parameter values of estimators.

✓ Completed | Go to next item